Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

2010-12

Photonic analog-to-digital converters preprocessing using the robust symmetrical number system for direct digitization of antenna signals

Tong, Kee Leong

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/4970

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

PHOTONIC ANALOG-TO-DIGITAL CONVERTER PREPROCESSING USING THE ROBUST SYMMETRICAL NUMBER SYSTEM FOR DIRECT DIGITIZATION OF ANTENNA SIGNALS

by

Kee Leong Tong

December 2010

Thesis Advisor:	Phillip E. Pace
Second Reader:	David C. Jenn

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** December 2010	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Photonic Analog-to-Digital Converter Preprocessing Using the Robust Symmetrical Number System for Direct Digitization of Antenna Signals		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Kee Leong Tong			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Center for Joint Services Electronic Warfare Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research, code 31, Washington DC		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE** A	

13. ABSTRACT (maximum 200 words)

The need to realize pervasive battlespace awareness is placing an increasing demand on the bandwidth and resolution performance of modern sensors, communication receivers and electronic warfare. Fundamental to realizing this demand is the omnipresent high-speed analog-to-digital converters. The need constantly exists for converters with lower power consumption. To reduce the number of power-consuming components, high-performance ADCs employ parallel configuration of analog folding circuits to symmetrically fold the input signal prior to quantization by high-speed comparators.

In this thesis, a prototype of an optical folding 6-bit ADC utilizing a 7-bit preprocessing architecture was implemented using the Robust Symmetrical Number System (RSNS). The RSNS preprocessing architecture is a modular scheme in which the integer values within each modulus (comparator states), when considered together, change one at a time at the next position i.e. Gray-code property. MATLAB simulations are used to help determine the properties of the RSNS. These properties include the dynamic range (largest number of distinct consecutive vectors) and the location of the dynamic range within the number system. Since the waveform repeats every fundamental period, a method that reduces all indexes to the 'lowest common denominator' is developed to find the symmetrical residues of each channel. Using the symmetrical residues determined, the corresponding DC shifts on each waveform can be calculated.

The architecture employs a three-modulus (mod 7, 8, 9) scheme to preprocess the antenna signal. Electro-optic modulation of the input signal to generate the required number of folds within the dynamic range was successfully carried out in the three-modulus realization using modulators with a small half-wave voltage. The detection output are carefully aligned and post-processed before amplitude analyzing with a high-speed comparator circuit responsible for the sampling and quantization of the signal (designed under a separate thesis). Low frequency analysis of the results using a 1 kHz input signal indicate a 5.42 effective number of bits (ENOB), a signal-to-noise ratio plus distortion (SINAD) of 34.42 dB, and a total harmonic distortion (THD) of –62.84 dB.

14. SUBJECT TERMS Robust Symmetrical Number System, Electro-optic modulation, Half-wave voltage, Effective number of bits (ENOB), Signal-to-ratio distortion (SINAD), Total harmonic distortion (THD)			**15. NUMBER OF PAGES** 105 **16. PRICE CODE**
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

PHOTONIC ANALOG-TO-DIGITAL CONVERTERS PREPROCESSING USING THE ROBUST SYMMETRICAL NUMBER SYSTEM FOR DIRECT DIGITIZATION OF ANTENNA SIGNALS

Kee Leong Tong
Major, Republic of Singapore Air Force
B.Eng., National University of Singapore, 2000

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2010

Author: Kee Leong Tong

Approved by: Phillip E. Pace
Thesis Advisor

David C. Jenn
Second Reader

R. Clark Robertson
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The need to realize pervasive battlespace awareness is placing an increasing demand on the bandwidth and resolution performance of modern sensors, communication receivers and electronic warfare. Fundamental to realizing this demand is the omnipresent high-speed analog-to-digital converters. The need constantly exists for converters with lower power consumption. To reduce the number of power-consuming components, high-performance ADCs employ parallel configuration of analog folding circuits to symmetrically fold the input signal prior to quantization by high-speed comparators.

In this thesis, a prototype of an optical folding 6-bit ADC utilizing a 7-bit preprocessing architecture was implemented using the Robust Symmetrical Number System (RSNS). The RSNS preprocessing architecture is a modular scheme in which the integer values within each modulus (comparator states), when considered together, change one at a time at the next position i.e. Gray-code property. MATLAB simulations are used to help determine the properties of the RSNS. These properties include the dynamic range (largest number of distinct consecutive vectors) and the location of the dynamic range within the number system. Since the waveform repeats every fundamental period, a method that reduces all indexes to the 'lowest common denominator' is developed to find the symmetrical residues of each channel. Using the symmetrical residues determined, the corresponding DC shifts on each waveform can be calculated.

The architecture employs a three-modulus (mod 7, 8, 9) scheme to preprocess the antenna signal. Electro-optic modulation of the input signal to generate the required number of folds within the dynamic range was successfully carried out in the three-modulus realization using modulators with a small half-wave voltage. The detection output are carefully aligned and post-processed before amplitude analyzing with a high-speed comparator circuit responsible for the sampling and quantization of the signal (designed under a separate thesis). Low frequency analysis of the results using a 1 kHz input signal indicate a 5.42 effective number of bits (ENOB), a signal-to-noise ratio plus distortion (SINAD) of 34.42 dB, and a total harmonic distortion (THD) of – 62.84 dB.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The need to realize pervasive battlespace awareness is placing an increasing demand on the bandwidth and resolution performance of modern sensors, communication receivers and electronic warfare. Fundamental to realizing this demand is the omnipresent high speed analog-to-digital converters (ADCs). The technological trend is to move the digital signal sampling up the receiving chain, i.e., sampling at the antenna. Sampling at the antenna eliminates the need for intermediate frequency and baseband processing. High speed ADCs that utilize photonic technology play a critical role in fulfilling this demand.

In this thesis, a prototype of an optical folding 6-bit ADC utilizing a 7-bit preprocessing architecture was implemented using the Robust Symmetrical Number System (RSNS). The architecture employs a three-modulus (mod 7, 8, 9) scheme to preprocess the antenna signal. The preprocessing scheme involves electro-optic modulation and signal alignment using three low half-wave voltage Mach-Zehnder Interferometers (MZIs), a continuous wave distributed feedback laser and three DC sources. To perform electro-optic modulation, an amplified electric field (antenna signal) is applied to each MZI to modulate the optical beam. Each antenna signal is amplified based on the required voltage to produce a modulated signal with a RSNS fundamental period. Signal alignment is completed by adding a DC signal via the DC port into the MZI. This critical process is necessary to achieve the proper alignment of the RSNS signals prior to sampling and conversion into the binary code.

After adjusting the alignment of the signals, each optical signal is converted to an electrical signal using three photodetectors. The detection output is post-processed before amplitude analyzing with a high-speed comparator circuit responsible for the sampling and quantization of the signal (designed under a separate thesis).

Low frequency analysis of the results using a 1 kHz input signal indicate a 5.42 effective number of bits (ENOB), a signal-to-noise ratio plus distortion (SINAD) of 34.42 dB, and a total harmonic distortion (THD) of – 62.84 dB.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Phillip E. Pace, for his guidance in the course of my research and implementation.

To Professor David C. Jenn, thank you for your guidance and assistance in proof-reading my thesis.

To Mr. James Calusdian, I humbly thank you for providing much of your knowledge about this research and for helping me integrate various parts of this system.

To my friends, Mr. Yean Wee Tan and Mr. Han Wei Lim, I also thank you both for always providing the help that I needed during the course of the thesis implementation.

Finally, to my wonderful and beautiful wife, Hwei Lian, thank you for your love, patience and continuous support. I share this accomplishment with you.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. ANALOG-TO-DIGITAL CONVERTERS

The need to realize pervasive battlespace awareness is placing an increasing demand on the bandwidth and resolution performance of modern sensors, communication receivers and electronic warfare. Fundamental to realizing this demand is the omnipresent high speed analog-to-digital converters (ADCs). The function of an ADC is to repetitively sample a time-varying waveform, usually at fixed time intervals, and generate a series of digital numbers to approximate the analog sample values [1]. Since the ADC is required to translate the sensor measurements into the digital language used by the computers, ADCs are one of the most widely used electronic component in signal processing systems.

The technological trend is to move the digital signal sampling up the receiving chain, i.e., sampling at the antenna. Sampling at the antenna eliminates the need for intermediate frequency and baseband processing. High speed ADCs that utilize photonic technology play a critical role in fulfilling this demand. These ADCs can digitize the RF signals directly at the antenna using Mach-Zehnder interferometers (MZIs) to amplitude modulate the RF into the optical domain. One of the first known photonic ADC techniques was introduced in 1979 by Taylor [1]. A 4-bit photonic ADC with Gray-code characteristics was implemented using four MZIs. This one bit per interferometer scheme was further improved with a different variation and demonstrated with a sampling rate of 1 gigasamples per second (GS/s) and a 2-bit resolution [2]. The downside to this implementation is the need for n MZIs for an n-bit resolution. This presented a significant limitation due to the increased power required to drive the devices in parallel.

B. PHOTONIC ANALOG-TO-DIGITAL CONVERTERS WITH UNIQUE ENCODING

A unique preprocessing technique is the time-stretch ADC presented by Jalali et al. [3–5]. A sampling rate of 130 GS/s has been demonstrated with ~7 effective number of bits (ENOB) using this technique. This is achieved by stretching an analog signal in the time domain to increase the effective sampling rate and the input bandwidth of ADC.

A different approach using an ultra-fast sample and hold circuit was reported by Urata et al. [6–9]. A 160 megasamples/s sampling rate with 3.5 ENOB was achieved using this design. The design uses low-temperature (LT)-grown GaAs metal-semiconductor-metal (MSM) photoconductive switches integrated with a Si-CMOS ADC. Samples are taken using a short-pulse laser to activate the switches and are held with the use of capacitors. Another ADC structure that uses 0.13-μm CMOS technology was presented by Wang and Liu. The ADC achieved 5 GS/s with a 4-bit ENOB for a 200-MHz input signal. This flash ADC design uses an interpolating architecture to reduce the number of amplifiers to enhance the bandwidth [10].

To effectively process the samples in a photonic sampled ADC, a photodiode must turn on and off rapidly [11]. Several proposals ranging from a moderate-speed ADC, time-division and wavelength division techniques have been proposed to ease the problem of ultra-fast sampling. Clark, Kang and Esman [12] introduced a scheme that uses a wavelength-interleaved photonic sampler. The use of wavelength-interleaved pulses achieved a ~7 bit ENOB with an 18-GHz modulator bandwidth limitation. The system uses a mode-locked laser and multiplexer to perform parallel digitization. Another demonstration using a 1-to-4 optical time-division demultiplexer achieved a high-linearity ADC with a 208 megasamples per sec (MS/s) sampling rate and 12 effective bits [13].

The ADCs that were discussed so far used a combination of optical and electronic devices. An optically sampled and quantized ADC concept has also been studied and simulated [14–15]. It employs a sample and hold technique using a tunable laser to

encode an electrical signal. The concept simulation reported a 4-bit ADC output. An all-optical ADC was proposed by Oda and Maruta [16]. A 2-bit optical quantizer was demonstrated using this all-optical scheme.

An integrated ADC utilizing silicon photonics technology offers higher sampling rates that electronic ADCs cannot accomplish. This may be realized by combining photonic and electronic devices in a single microchip [17].

C. FOLDING ADC PREPROCESSING ARCHITECTURE: SYMMETRICAL NUMBER SYSTEMS

The need constantly exists for converters with higher resolution, faster conversion speeds and lower power dissipation. To reduce the number of power-consuming components, high-performance ADCs employ a parallel configuration of analog folding circuits to symmetrically fold the input signal prior to quantization by high-speed comparators (analog preprocessing) [18]. Pace et al. reported in 1995 the experimental transfer characteristics of a three-channel 5-bit symmetrical number system (SNS) guided-wave ADC [19] and an 8-bit integrated optical SNS ADC [20]. The SNS preprocessing provides resolution greater than 1 bit per interferometer. The results demonstrated the feasibility of the SNS ADC concept.

Three symmetrical number systems formulations, i.e., SNS, the optimum symmetrical number system (OSNS), and the robust symmetrical number system (RSNS) have been used previously to increase the efficiency of folding ADC converter architectures, efficiently encode digital antenna links, and increase the resolvable bandwidth of two- and three- channel digital intercept receivers [21–28]. Each symmetrical number system is composed of a number of pairwise relatively prime moduli. The integers within each modulus m_i are derived from a symmetrically folded waveform. Due to the presence of ambiguities, the set of integers within each symmetrical number system modulus does not form a complete residue system. The ambiguities are resolved by considering the combined values from all channels. The SNS is used to preprocess the analog signal reducing the number of amplitude analyzing comparators that are required. The dynamic range of the SNS is $\hat{M}_{SNS}$. The OSNS is

defined to extend the dynamic range of the SNS [21]. By recombining the *N* channels, the OSNS is rendered a complete system having a one-to-one correspondence with the residue number system (RNS) with dynamic range $M = \Pi m_i > \hat{M}_{SNS}$ [29]. The SNS and the OSNS, however, have the encoding error characteristics like that of the RNS. Encoding errors can occur when the input signal lies about any code transition point. That is, if one integer within the modulus is incorrect, the resulting value that is decoded has a large error [29].

The RSNS scheme is a modular scheme in which the integer values within each modulus, when considered together, change one at a time at the next position (Gray code properties). This eliminates the possible encoding errors that may result in addition to extending the resolution beyond 1-bit per interferometer. Calculation of the dynamic range and its position within the combined sequences is necessary for any application. Styer and Pace defined the two-channel RSNS and presented a theorem that gives its dynamic range for relatively prime moduli $m_1,\ m_2,\ 5 \le m_1 \le m_2$ [29].

Pace et al. presented a folding ADC preprocessing architecture employing a RSNS with Gray-code properties [18]. Closed-form expressions for the dynamic range are also presented for channel moduli of the form $m_1 = 2^k - 1,\ m_2 = 2^k,\ m_3 = 2^k + 1$. The paper highlighted that the selection of the shifts and their permutations among the three moduli have no effect on the dynamic range, $\hat{M}_{RSNS}$. However, the beginning and ending points are different. For any particular channel, the folding waveform crosses the comparator thresholds in turn every *N* LSBs. Within the ADC, however, only one comparator threshold is crossed at any LSB code transition point (Gray-code property). The paper noted that, although the dynamic range of the RSNS ADC is not optimum, the absence of encoding errors makes the RSNS encoding scheme an efficient approach for folding ADC designs.

M. Arvizo in her Masters' Thesis demonstrated a full implementation of a 6-bit RSNS ADC concept (mod 3, 4, 5) by combining photonic and electronic technologies [30]. The architecture employs a three-modulus scheme to preprocess the antenna signal. An analysis of the results using a 20-kHz input signal indicate a 5.33 effective number of

bits (ENOB), a signal-to-noise plus distortion (SINAD) of 33.8 dB, and a total harmonic distortion (THD) of −43 dB. The importance of accurate electro-optic modulation and signal alignment in order to obtain accurate folding waveforms was noted in the thesis. The input power limitation of the RF port became the basis of using the DC bias port instead. With a summing amplifier, the antenna and DC signal were both injected into the DC port. The DC port, however, has a 40-kHz bandwidth limitation. Although the encoding scheme had minimal errors with low-frequency input signal, the errors increased as the modulated signals became distorted at higher frequencies.

D. PRINCIPAL CONTRIBUTIONS

In this thesis, a prototype of an optical folding 7-bit preprocessing architecture was designed using the Robust Symmetrical Number System (RSNS). Due to an inconsistent extinction ratio of the modulators, a workaround was proposed and a 6-bit ADC utilizing the 7-bit design was implemented instead. The architecture employs a three-modulus (mod 7, 8, 9) scheme to preprocess the antenna signal.

A literature search was conducted on the work that has been done. Theoretical calculations were performed to specify the parameters of the hardware components that had to be purchased. MATLAB simulations were carried out on the feasibility and expected results of the proposed design architecture. A formula that reduces the beginning and ending vectors to small index values is used to determine the symmetrical residues of each channel. This provides a fast and easy way to find the integer values of the position index for each channel.

The design efficiently couples an RF signal into the optical domain using a parallel management of three MZIs. The RF signal is preprocessed using an arrangement of three power amplifiers and attenuators. The optical to electrical conversion is performed by three photodetectors.

The three power amplifiers, fixed attenuators and variable attenuators were designed to provide sufficient gain and input voltage to the MZIs to fully exploit the bandwidth range of the MZIs and to accurately preprocess the antenna signal. Due to limitation of the high speed comparators [31] during the integration phase, the design was

changed, and three function generators producing three coherent waveforms had to be used to supply the required voltage. For the amplitude modulation stage, the three low half-wave voltage, wideband MZIs were interfaced with a continuous wave distributed feedback (DFB) laser using a 1×3 splitter. The preprocessed signal was then applied to the RF ports of the interferometers to amplitude modulate the continuous wave laser signal. A DC signal was applied on the DC bias port of each interferometer to shift and control the alignment of the folded signals. An InGaAs photodetector was connected at the output of each modulator to convert the modulated optical signal from each interferometer into an electrical signal. The detection outputs are carefully aligned and post-processed [30] before amplitude analyzing with a high-speed comparator circuits which sample and quantize the detector output signals (designed under a separate thesis [31]). Low frequency analysis of the results using a 1-kHz input signal indicate a 5.42 effective number of bits (ENOB), a signal-to-noise ratio plus distortion (SINAD) of 34.42 dB, and a total harmonic distortion (THD) of – 62.84 dB.

E. THESIS OUTLINE

The RSNS theory and the steps involved in defining the dynamic range, determining the integer values of the position index for each channel and locating the position index on the RSNS waveform are explained in Chapter II.

A description of how the photonic ADC architecture front-end was designed and implemented is contained in Chapter III.

The characterization, analysis and evaluation of the hardware components and preprocessing architecture results are explained in Chapter IV.

The measurement results, calculations and a discussion of the important parameters that describe the ADC's performance are presented in Chapter V.

In Chapter VI, the conclusions and suggestions for future research are presented.

II. ROBUST SYMMETRICAL NUMBER SYSTEM

In this chapter, there is an explanation of the basic theory of the RSNS and how the dynamic range for a 3-channel (Mod 7, 8, 9) RSNS is determined. Since the waveform repeats every fundamental period, a formula that reduces the beginning and ending vectors to small index values is used to determine the symmetrical residues of each channel. With the location of the symmetrical residues determined, the corresponding shifts on each waveform can be calculated. This forms the preparatory work needed to support the design of the hardware requirements.

A. RSNS THEORY

In the RSNS, *N* different periodic symmetrical waveforms are used to generate a series of integers based on the following sequence [18]:

$$x_{m(N)} = [0,1,2,...m-1,m,m-1,...,2,1] \tag{2.1}$$

where $x_{m(N)}$ is a row vector and m is a pairwise relatively prime modulus and is a positive integer greater than 0. For an *N*-channel RSNS with $N = 3$, the sequence is [18]

$$x_{m(N)} = [0,0,0,1,1,1,2,2,2,...m,m,m...2,2,2,1,1,1]. \tag{2.2}$$

In each sequence, each value from row vector $x_{m(N)}$ is repeated N times. This results in a periodic sequence with a period of [18]

$$P_{RSNS} = 2mN\ . \tag{2.3}$$

The discrete RSNS integrers for a single channel are [18]

$$g[n] = \begin{cases} \left\lfloor \dfrac{n - s_{shift_i}}{N} \right\rfloor, & s_{shift_i} \le n \le Nm_i + s_{shift_i} + 1 \\ \left\lfloor \dfrac{2nm_i + N - n + s_{shift_i} - 1}{N} \right\rfloor, & Nm_i + s_{shift_i} + 2 \le n \le 2Nm_i + s_{shift_i} - 1 \end{cases} \tag{2.4}$$

where g is the n^{th} term of channel i and m_i is the channel modulus. The sequence shifts s_{shift_i} have values $s_{shift_i} \equiv 0,1,2,3,...,N-1(\text{mod}\, N)$, and the number of channels is $N \geq 2$. Therefore, an *N*-channel RSNS contains *N*-row vectors with sequence values based on (2.2). The discrete states of the RSNS are indexed using the $s_{shift_i} = 0$ row vector with the index starting from the first zero. The RSNS has a fundamental period of [18]

$$PF_{RSNS} = [2m_1 N, 2m_2 N, ..., 2m_N N] = 2N[m_1, m_2, m_3] \tag{2.5}$$

where $[a_1, a_2, ..., a_N]$ with least common multiple of $a_1, a_2, ..., a_N$.

The RSNS is compatible with symmetrical folding waveforms (with folding period $2m_i$) where the integers within each sequence are determined by a proper placement of threshold values to amplitude analyze the folding waveform. The three-channel (mod 7, 8, 9) RSNS generated (for h = 0 to 128) with the calculated threshold values for the comparators using MATLAB (see Appendix E) are as shown in Figures 1, 2 and 3. Each folding waveform folds at $2Nm_i$. The corresponding symmetrical residue representation for h = 0 to 36 is as shown in Table 1.

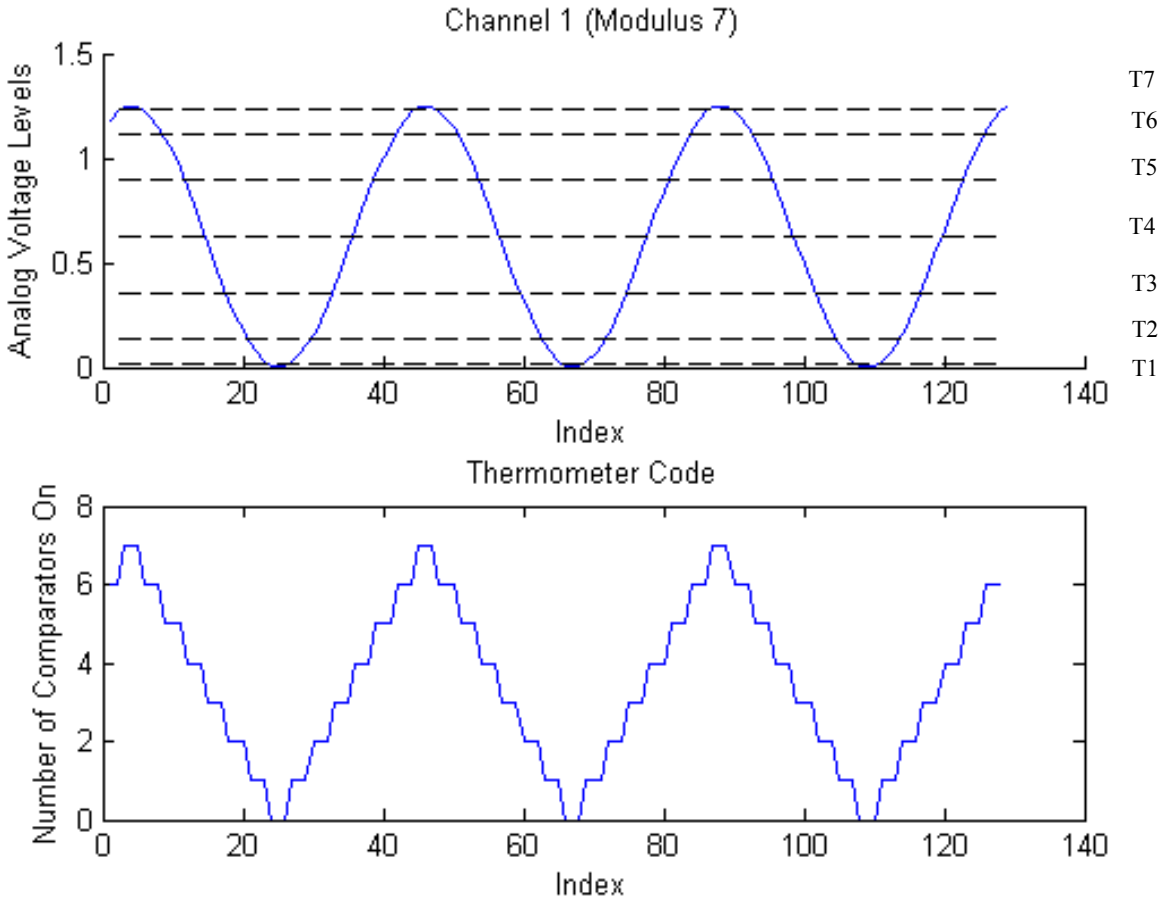

Figure 1. RSNS waveform and thermometer code (left shift), mod 7, $s_{shift_i} = [0\ 1\ 2]^T$.

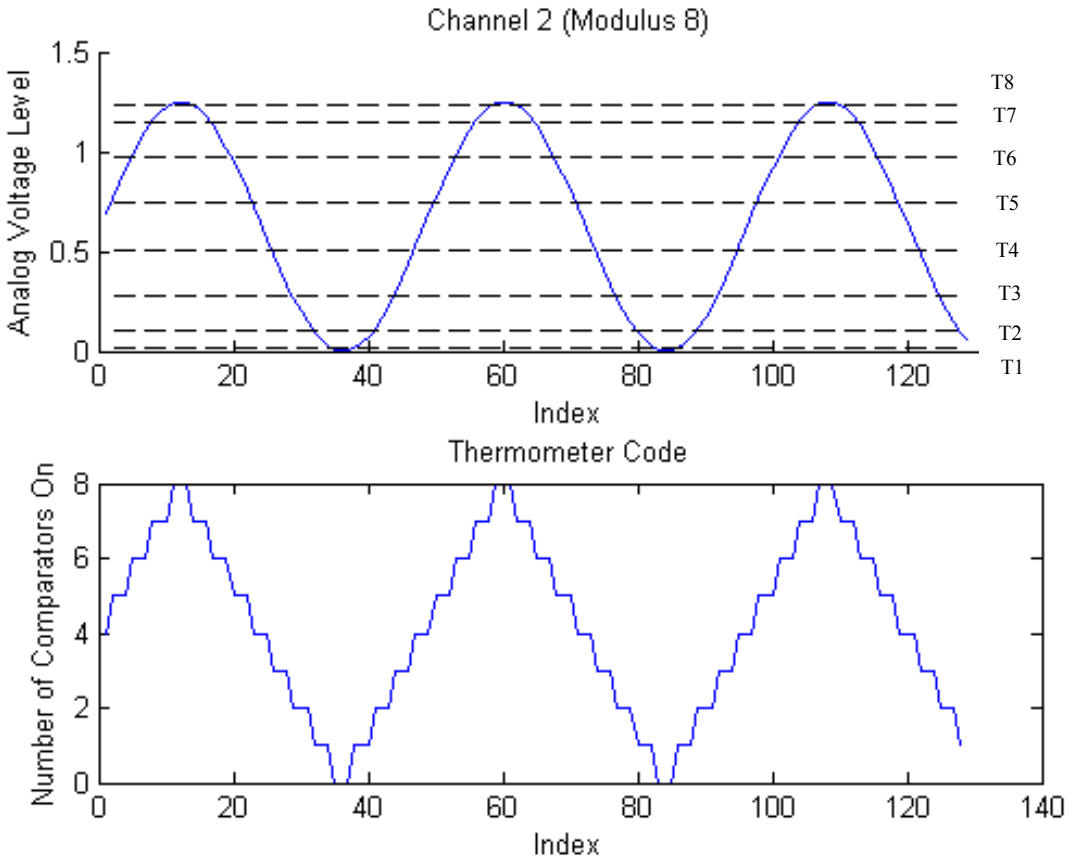

Figure 2. RSNS waveform and thermometer code (left shift), mod 8, $s_{shift_i} = [0\ 1\ 2]^T$.

Each of the integer values within the sequence are shown based on (2.2). The integer value in each sequence is generated by comparing the amplitude of the symmetrical folding waveform to the m_i comparator threshold values. To ensure that the sequence in (2.2) is followed, the threshold values are set such that the interval between two thresholds will generate integers that are repeated N times. For example, the amplitude of the $m_1 = 7$ signal is less than the seven threshold values T1 to T7 at position index $h = 0$. Hence, each comparator has an output that corresponds to an integer value of 0 as shown in Table 1. At index position $h = 3$, the amplitude of the signal is greater than threshold T1, and the output of the corresponding comparator will change to 1. As expected, the next integer value change occurs at position index $h = 6$. This follows the

sequence pattern of having *N* repeated integers in each row vector. As the signal reaches index position $h = 21$, the amplitude of the signal exceeds all seven threshold values. Each of the seven comparators is on and an integer value of 7 is obtained. The sequence continues until the length of the vector reaches the maximum system dynamic range $\hat{M} = 133$ for the N = 3 ($m_1 = 7, m_2 = 8, m_3 = 9$). The maximum system dynamic range is the number of consecutive column vectors that are distinct.

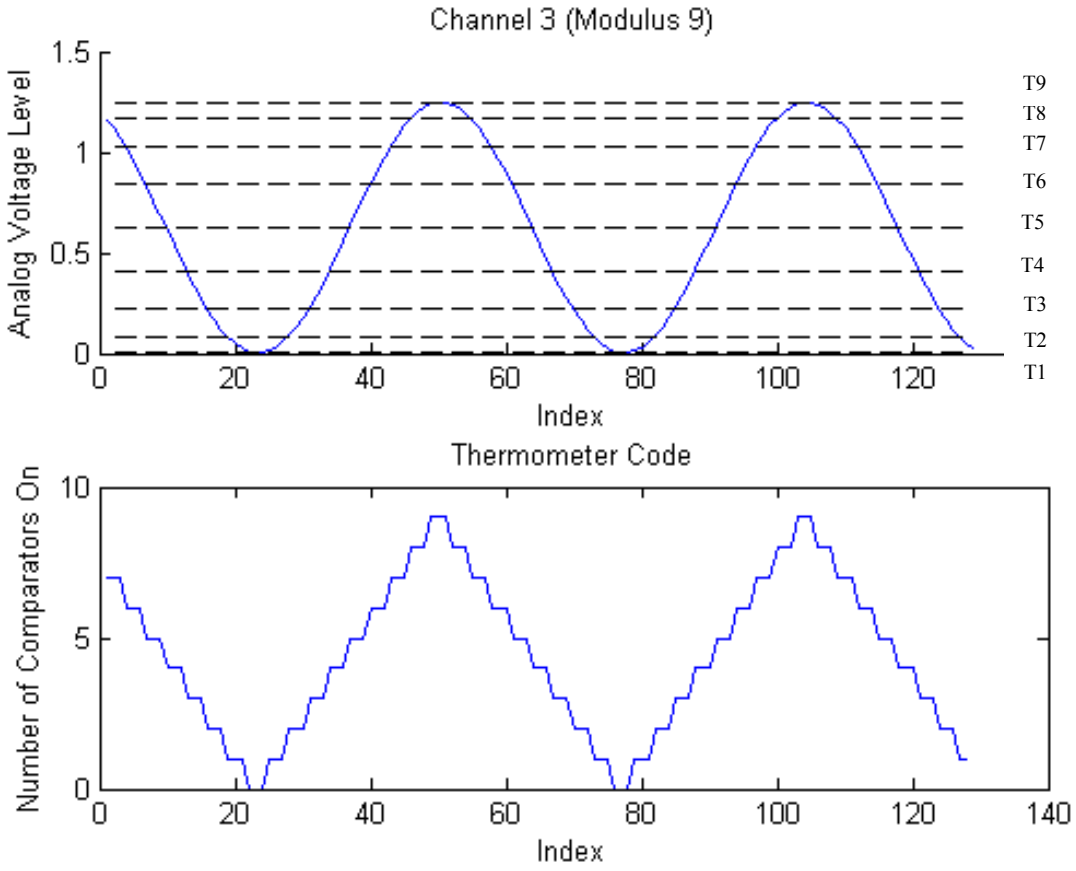

Figure 3. RSNS waveform and thermometer code (left shift), mod 9, $s_{shift_i} = [0\ 1\ 2]^T$.

The same process is used to generate the integers contained in $m_2 = 8$ and $m_3 = 9$ row vectors. To maintain the characteristic of the sequence in (2.2), the threshold values must be set correctly. The key to maintaining *N* repeated integers in each sequence is

having a precise set of threshold values coupled with the three accurately aligned folding waveforms. An illustration of the modulated signals with the threshold values are explained further in Chapter III.

From Table 1, the Gray-code property of RSNS is demonstrated by observing the integer values in each column vector. Observe that for each code transition, only one integer value changes within a column vector. With this property, the RSNS scheme provides better error control during data processing.

Table 1. An illustration of a three-channel RSNS (left shift).

mod 7	0	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	…
mod 8	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	…
mod 9	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	…
h	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	…

mod 7	6	6	7	7	7	6	6	6	5	5	5	4	4	4	3	3	3	2	…
mod 8	6	7	7	7	8	8	8	7	7	7	6	6	6	5	5	5	4	4	…
mod 9	7	7	7	8	8	8	9	9	9	8	8	8	7	7	7	6	6	6	…
h	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	…

B. DETERMINATION OF THE DYNAMIC RANGE

The system dynamic range ($\hat{M}$) of the RSNS is the maximum number of distinct vectors without an ambiguity. In any RSNS application it is important to determine $\hat{M}$ in order to determine which combination of moduli form the best RSNS with $\hat{M} \geq 2^p$, where p = number of bits needed for the ADC. When three channels are considered and the channel moduli are of the form $m_1 = 2^k - 1,\ m_2 = 2^k$ and $m_3 = 2^k + 1$, the closed form expression for $\hat{M}$ is

$$\hat{M} = \frac{3}{2} m_1^2 + \frac{15}{2} m_1 + 7 \tag{2.6}$$

where $m_1 \geq 3$ [18].

From (2.6), the system dynamic range calculated for mod (7, 8, 9) is 133. The key to generating and encoding the RSNS into an EO folding architecture is having a correct input voltage to the MZI, V_{RF_i} and proper alignment of signals. The calculations performed in the design of the preprocessing architecture are documented in Chapter III. The starting and ending position vectors for a right and left shift implementation are found using two MATLAB codes as enclosed in Appendix A and Appendix B, respectively. The right shift code employs a naïve approach. This approach is more computationally intensive as compared to the left shift algorithm, which reduces the computational time by several orders of magnitude. In addition, the efficient algorithm uses far less memory than the naïve algorithm and, therefore, can find the *N*-sequence $\hat{M}$ and position for moduli sets with much larger fundamental periods [32].

The outputs from both codes are as shown in Tables 2 and 3. It can be seen that the point indices corresponding to $\hat{M}$ (beginning point and ending point) are different. This represents the useful range of the RSNS that can be used to encode the detected modulation outputs. The three channels need to be properly aligned relative to each other based on the beginning and ending position of the position index prior to sampling.

Table 2. System Dynamic Range for Mod (7, 8, 9) RSNS (right shift).

Moduli	**shift**	**System dynamic range**	**Fundamental Frequency**	**Beginning-Ending of system dynamic range**
7 8 9	0 1 2	133	3024	649-781

Table 3. System Dynamic Range for Mod (7, 8, 9) RSNS (left shift).

Moduli	**shift**	**System dynamic range**	**Fundamental Frequency**	**Beginning-Ending of system dynamic range**
7 8 9	0 1 2	133	3024	733-865

C. DETERMINING THE INTEGER VALUES OF THE POSITION INDEX FOR EACH CHANNEL

The beginning and ending points of system dynamic range for the mod (7, 8, 9) RSNS with their corresponding shift are summarized in Tables 2 and 3. Since the waveform for each channel repeats every fundamental period, a method that subtracts out integer multiples of the $2Nm_i$ codes within a folding period is developed to find the symmetrical residues of each channel. The formula is

$$h_{m_i} = h - \left\lfloor \frac{h}{2Nm_i} \right\rfloor 2Nm_i \tag{2.7}$$

where h_{m_i} is the index value h of mod m_i. An example of how the indexes can be derived quickly and accurately without having to map out the entire sequence manually is as shown in Table 4. From (2.7), $h_7 = 20$, $h_8 = 14$, $h_9 = 8$. Verification with Table 4 shows that the symmetrical residues determined are all present at $h = 62$. This provides a fast and easy way to determine the symmetrical residues at the beginning position index for each channel and is especially useful when the position index is a few hundred positions away. Positive identification of the position index is important as this will enable a quick calculation of the bias voltage value needed for each channel to shift the waveform such that the threshold crossings give the integer values within the sequence at a chosen input voltage.

Table 4. Mod (7, 8, 9) RSNS partial structure with $s_{shift_i} = [0\ 1\ 2]^T$.

mod 7	0	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	…
mod 8	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	…
mod 9	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	…
h	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	…

mod 7	6	6	7	7	7	6	6	6	5	5	5	4	4	4	3	3	3	2	…
mod 8	6	7	7	7	8	8	8	7	7	7	6	6	6	5	5	5	4	4	…
mod 9	7	7	7	8	8	8	9	9	9	8	8	8	7	7	7	6	6	6	…
h	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	…

mod 7	2	2	1	1	1	0	0	0	1	1	1	2	2	2	3	3	3	4	…
mod 8	4	3	3	3	2	2	2	1	1	1	0	0	0	1	1	1	2	2	…
mod 9	5	5	5	4	4	4	3	3	3	2	2	2	1	1	1	0	0	0	…
h	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	…

mod 7	4	4	5	5	5	6	6	**6**	7	7	7	6	6	6	5	5	5	4	…
mod 8	2	3	3	3	4	4	4	**5**	5	5	6	6	6	7	7	7	8	8	…
mod 9	1	1	1	2	2	2	3	**3**	3	4	4	4	5	5	5	6	6	6	…
h	55	56	57	58	59	60	61	**62**	63	64	65	66	67	68	69	70	71	72	…

D. LOCATION OF POSITION INDEX ON THE RSNS WAVEFORM

The location of the beginning and ending position index and their respective integer values are determined as detailed in section B and C. To translate this into voltage shift values, the least significant bit (LSB) needs to be calculated. The LSB represents the minimal DC bias voltage value to be applied to the MZI for every shift in the position index. The size of the LSB within each of the *N* channels is the same. From (2.5), there are $P_{m_i} = 2Nm_i$ LSBs within a complete fold $2V_\pi$ resulting in

$$LSB = \frac{V_{\pi_i}}{Nm_i} \text{ V} \tag{2.8}$$

where V_{π_i} is the effective half wave voltage the MZI.

A complete mapping of the position vectors for left shift $[0\ 1\ 2]^T$ for the dynamic range from *h* = 733 to 860 is shown in Table 5. The starting position is *h* = 733 for a 7-bit

(2^7=128) implementation. From (2.7) for h = 733, from Table 1, $h_7 = 19$ $h_8 = 13$ and $h_9 = 31$. The DC bias voltage required for channel 1, 2 and 3 are 19, 13 and 31 LSBs away, respectively.

Table 5. Mod (7, 8, 9) RSNS, $s_{shift_i} = [0\ 1\ 2]^T$, dynamic range h=733 to 861.

mod 7	6	**6**	6	7	7	7	6	6	6	5	5	5	4	4	4	…
mod 8	4	**4**	5	5	5	6	6	6	7	7	7	8	8	8	7	…
mod 9	8	**7**	7	7	6	6	6	5	5	5	4	4	4	3	3	…
h	732	**733**	734	735	736	737	738	739	740	741	742	743	744	745	746	…

mod 7	3	3	3	2	2	2	1	1	1	0	0	0	1	1	1	…
mod 8	7	7	6	6	6	5	5	5	4	4	4	3	3	3	2	…
mod 9	3	2	2	2	1	1	1	0	0	0	1	1	1	2	2	…
h	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	…

mod 7	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	…
mod 8	2	2	1	1	1	0	0	0	1	1	1	2	2	2	3	…
mod 9	2	3	3	3	4	4	4	5	5	5	6	6	6	7	7	…
h	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	…

mod 7	7	7	7	6	6	6	5	5	5	4	4	4	3	3	3	…
mod 8	3	3	4	4	4	5	5	5	6	6	6	7	7	7	8	…
mod 9	7	8	8	8	9	9	9	8	8	8	7	7	7	6	6	…
h	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	…

mod 7	2	2	2	1	1	1	0	0	0	1	1	1	2	2	2	…
mod 8	8	8	7	7	7	6	6	6	5	5	5	4	4	4	3	…
mod 9	6	5	5	5	4	4	4	3	3	3	2	2	2	1	1	…
h	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	…

mod 7	3	3	3	4	4	4	5	5	5	6	6	6	7	7	7	…
mod 8	3	3	2	2	2	1	1	1	0	0	0	1	1	1	2	…
mod 9	1	0	0	0	1	1	1	2	2	2	3	3	3	4	4	…
h	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	…

mod 7	6	6	6	5	5	5	4	4	4	3	3	3	2	2	2	…
mod 8	2	2	3	3	3	4	4	4	5	5	5	6	6	6	7	…
mod 9	4	5	5	5	6	6	6	7	7	7	8	8	8	9	9	…
h	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	…
mod 7	1	1	1	0	0	0	1	1	1	2	2	2	3	3	3	…
mod 8	7	7	8	8	8	7	7	7	6	6	6	5	5	5	4	…

mod 9	9	8	8	8	7	7	7	6	6	6	5	5	5	4	4	…
h	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	…

mod 7	4	4	4	5	5	5	6	6	6	7	7	7	6	6	6	…
mod 8	4	4	3	3	3	2	2	2	1	1	1	0	0	0	1	…
mod 9	4	3	3	3	2	2	2	1	1	1	0	0	0	1	1	…
h	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	…

E. SUMMARY

In this chapter, the RSNS theory was explained. A detailed discussion of the preparatory work to be done prior to designing the preprocessing architecture was presented. In the next chapter, the design and implementation of a photonic RSNS ADC preprocessing architecture leveraging on the preparatory work done in Chapter II is discussed. Design specifications, calculations and hardware selection will be highlighted and discussed in detail.

THIS PAGE INTENTIONALLY LEFT BLANK

III. RSNS ADC ARCHITECTURE DESIGN

A background on RSNS theory was presented in Chapter II, and a detailed discussion of the preparatory work to be done prior to designing the preprocessing architecture was presented. This included determining the dynamic range, identifying and locating the symmetrical residues and determining the corresponding shifts required for each channel so that the three channels can be aligned in such a way that the Gray-code properties can be extracted.

The implementation of a RF photonic front-end ADC preprocessing architecture is explained in this chapter. The components selection criteria and effects of each device's operating characteristics and limitations on the system operation are pointed out.

A. RF PHOTONIC ADC SYSTEM DESIGN

In [30], a full implementation of a 6-bit RSNS ADC concept was constructed by combining photonic and electronic technologies. The architecture employs a three-modulus (3, 4, 5) scheme to preprocess the antenna signal. In this thesis, an improved architecture employing a three-modulus (7, 8, 9) scheme to preprocess the antenna signal is implemented and tested. Analysis of results demonstrates an improvement in the dynamic range (from 43 to 133) and reinforces the feasibility of using electro-optic devices to fully implement an optical folding ADC using the RSNS concept.

The photonic ADC front-end prototype was built by combining photonic and electronic devices. It was fully implemented by combining several signal processing stages to produce three aligned electrical signals. These signals are then passed through the RSNS-to-binary conversion stage [31]. Component selection for each signal processing phase are addressed in subsequent sections.

The block diagram of the RF photonic ADC architecture is shown in Figure 4. An RF signal received by the antenna (in this case a signal generator) is split into three channels and amplified individually to the maximum voltage required to generate the folding periods. A 1.55-μm continuous wave optical signal from a distributed feedback (DFB) laser is used after going through a 1×3 splitter. The splitter divides the laser signal

into three beams, providing a signal for each Mach-Zehnder modulator. As the amplified RF signal is applied to the electrodes of the modulator, amplitude modulation of the laser signal occurs. To obtain the necessary phase shift for alignment of the RSNS channels a DC voltage is also applied on the bias port of each modulator. After amplitude modulation of the laser, each modulated optical signal is sent to a photodetector to convert it into an electrical signal. The electrical signal is amplified and further processed using the RSNS-to-binary scheme utilizing an FPGA [31]. The encoding scheme converts the three-modulus RSNS signals to a bipolar 7-bit binary code. Each code within the 7-bit binary representation corresponds to a quantized value (dynamic range $\hat{M} = 128$).

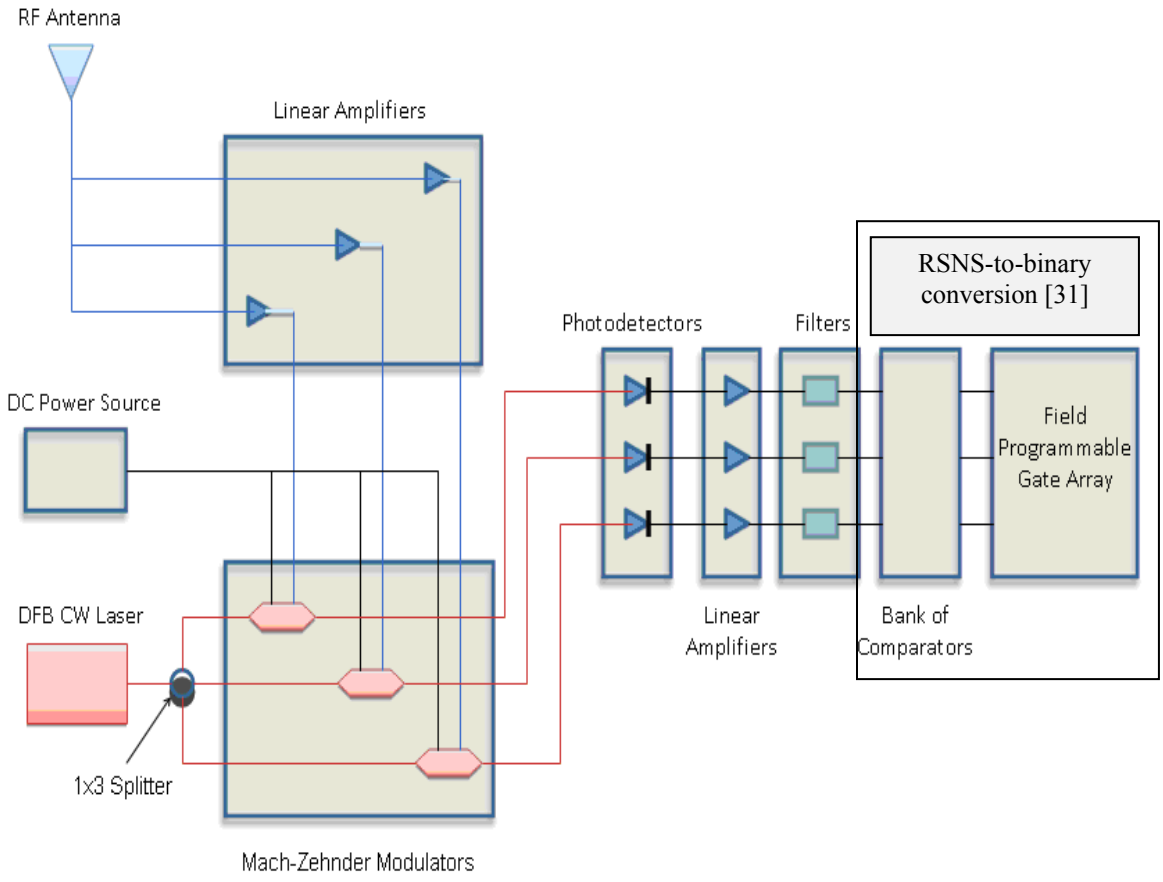

Figure 4. Block diagram of RSNS based photonic ADC architecture (From [30]).

B. RF SIGNAL AMPLIFICATION

Each amplifier has a variable gain to obtain the required voltage for electro-optic modulation. For a three-modulus RSNS structure, each channel requires a different voltage to obtain the following fundamental periods [18]

$$PF_{RSNS} = 2N[m_1, m_2, m_3] \tag{3.1}$$

for $[m_1, m_2, m_3] = [7, 8, 9]$ where [.] is the LCM operator = πm_i and $N = 3$ $PF_{RSNS} = 3024$. The fundamental frequency for each RSNS channel can be represented as

$$f_{RSNS_i} = \frac{1}{P_{RSNS}} = \frac{1}{2Nm_i} \tag{3.2}$$

where $i = [1, 2, 3]$ is the channel index.

From the relationship between the half-wave voltage of the MZ modulator and the period in each channel, the required applied voltage for a bipolar ADC using the RF port is [18]

$$V_{RF_i} = \frac{\hat{M}}{2Nm_i} V_{\pi_i} \tag{3.3}$$

where V_{π_i} is the effective half-wave voltage. From here we can see that the most important parameter is the V_π of each modulator. With a smaller V_π the required signal amplification V_{RF_i} will be lower.

The voltage and power ratings for the three-modulus (7, 8, 9) channels using (3.3) is as shown in Table 6. It can be seen that for mod 7, the highest power rating is 29 dBm. Although this is higher than the modulator's specification of 27 dBm, it is a significant improvement from the 33.5 dBm requirement in the three-modulus (3,4,5) implementation [30] where the high input power, the DC port had to be used instead of the RF port. This lowered the bandwidth processing capability of the setup. It also meant that the alignment of the three channels via the application of the bias shift voltages had to be coupled to the V_{RF_i} via a summing amplifier [30]. This posed a challenge in

adjusting the bias voltage to be applied to each channel when performing the channel alignment in the laboratory. With the improved V_π , the RF port was used and a signal of 1-kHz was successfully preprocessed and encoded. The availability of the DC port allowed the required bias voltage to be applied on the bias port of each modulator to obtain the necessary phase shift in the transmisivity for alignment of the RSNS channels.

Table 6. Calculated applied RF port voltage and power requirements.

i^{th} Channel	RF Port Half-wave Voltage $V_{\pi RF_i}$	Calculated RF Port Applied Voltage V_{RF_i}	Power Rating
1 (mod 7)	3 V	9.14 V	29 dBm
2 (mod 8)	3 V	8.0 V	28 dBm
3 (mod 9)	3 V	7.1 V	27 dBm

With the required voltage and power rating, it helped to narrow down the required amplifier and attenuators needed for the signal amplification design. Due to the unavailability of Low Noise Amplifiers (LNA) with power output of at least 0.5 W, a power amplifier with a low noise figure is proposed. To cater for the differing V_{RF_i} requirements for the three channels, a flexible and adjustable amplification circuit had to be employed. The amplification design for three channels is as shown in Figure 5.

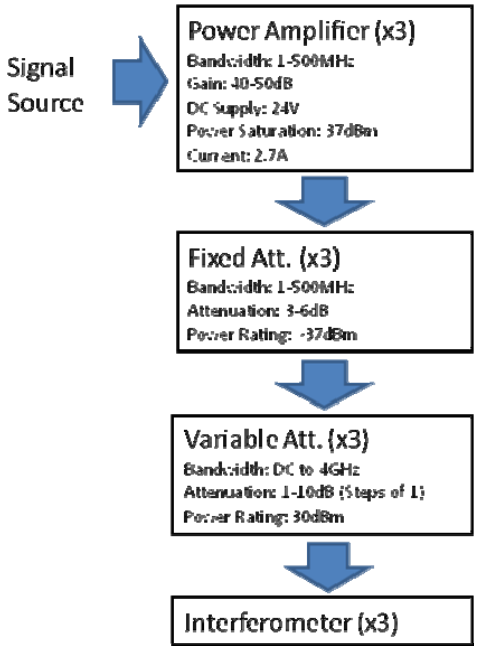

Figure 5. RF preprocessing circuit design.

The mixture of fixed and variable attenuators allow the tuning of the voltage and power values prior to the input of each modulator. A summary of the components selected is as shown in Table 7.

Table 7. Signal amplification components.

Component	Model	Remarks
Power Amplifier	Nextec-RF (NB00415)	Amplification of signal source
Fixed Attenuator	Bird Technologies (100-A-MFA-03/06)	Provide attenuation to suitable levels for input to the variable attenuator (3 dB and 6 dB)
Variable Attenuator	Agilent 8494A	Fine tune V_{RF_i} required for each channel

C. MODULATION AND RSNS SIGNAL ALIGNMENT

Electro-optic modulation is achieved using an MZI. An electric field is used to amplitude modulate the light from an optical source such as a laser. The basic structure of an electro-optic modulator is as shown in Figure 6. A basic structured modulator [33] is comprised of 1) two waveguides, 2) two Y-junctions and 3) RF/DC electrodes. Optical signals coming from the laser are launched into the modulator through the polarization maintaining (PM) fiber. It is then equally split into two optical waveguides by the first Y-junction on the substrate. When the voltage is not applied to the RF electrode, the two signals are re-combined coherently at the second Y-junction. In this case, the output signal from the modulator is recognized as "ONE." When the voltage V_π is applied to the RF electrode, due to the electro-optic effects of the modulator substrate, the refractive index is changed, and the propagation constant of the optical signal in one arm is delayed with respect to the other arm. As the two signals are re-combined at the second Y-junction, they are transformed into higher order mode and lost as a radiation mode. When two signals are recombined out of phase, the output signal from the modulator is recognized as "ZERO." The voltage difference which induces this "ZERO" and "ONE" is called the half-wave voltage V_π of the modulator.

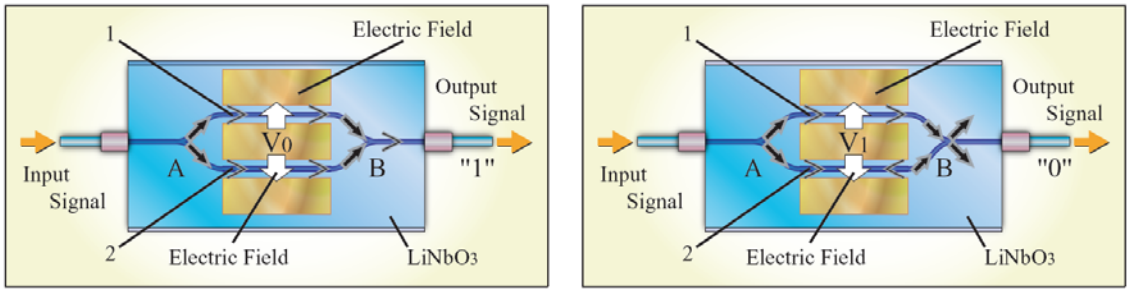

Figure 6. Basic structure of a MZI (From [34]).

For this system, a DFB laser is the optical source. It provides a 1.55 μm continuous wave signal. A laser diode controller module is used to control the intensity. Injecting 150 mA of current into the laser diode, we produce enough intensity for the photodetector to detect.

As the optical beam travels through the crystal lattice of the modulator, an applied electric field causes amplitude modulation on the light beam. The applied electric field must have adequate amplitude to produce the essential RSNS period or frequency for each channel. The intensity of the MZ modulator output is defined as [35]

$$I = I_0 \cos^2\left(\frac{\pi V}{2V_\pi}\right) \tag{3.4}$$

where I_0 is the input optical intensity, V is the applied voltage, and V_π is the half-wave voltage of the MZ modulator.

The transfer characteristic or transmissivity of an optical modulator, each with a different half-wave voltage is demonstrated in Figure 7 using (3.4) [30]. The plots depict how the period of the modulated optical beam varies as a function of applied voltage. As the applied voltage increases, the period of the modulated signal decreases. Also, each modulated signal is symmetric to the y-axis, and the peaks occur when the applied voltage is equal to an integer multiple of $2V_\pi$. Adding a DC voltage into the modulator causes a phase shift in the modulated signal. In such case, the peaks will also shift left and right depending on the applied DC voltage. In optical communications, it is convenient to set the bias point at 50% transmission (quadrature) point, as shown in Figure 8. The electrical signals are transformed into optical digital signal by electrically switching the voltage from $V_B - \frac{V_\pi}{2}$ to $V_B + \frac{V_\pi}{2}$.

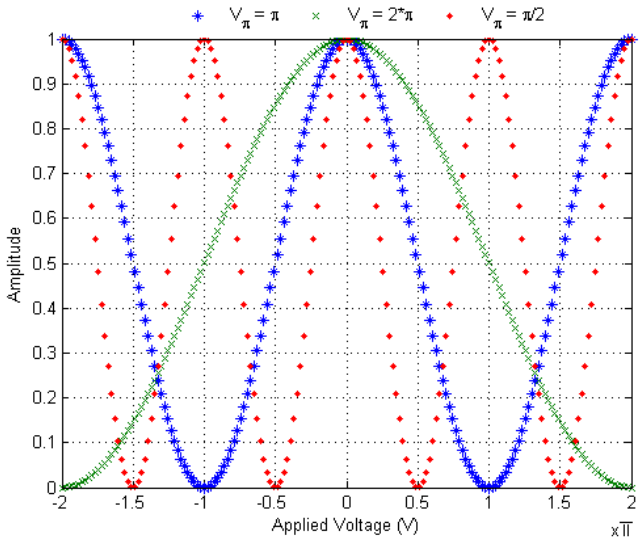

Figure 7. Transfer function of a Mach-Zehnder modulator (From [30]).

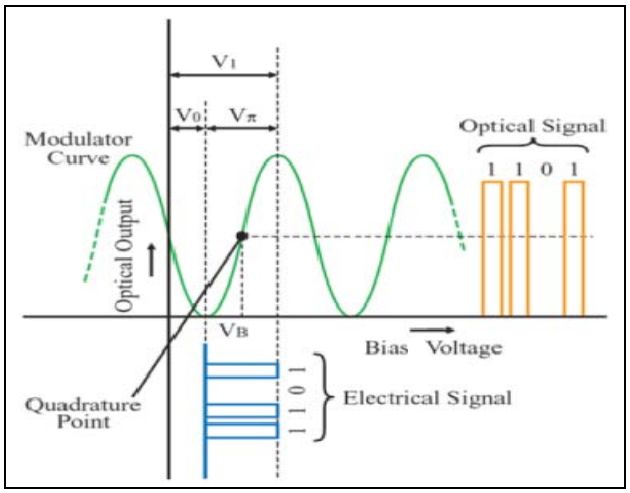

Figure 8. Electrical-optical plot (From [36]).

The key performance parameters of the modulator chosen for this thesis are as shown in Table 8. The key to encoding the RSNS into an EO folding architecture is having a correct V_{RF_i} and proper alignment of the modulation signals. Recall Table 2 summarizes the beginning-ending of system dynamic range for the three modulus RSNS with their corresponding shift.

Table 8. Modulator key performance parameters.

Model	Operating Wavelength	Insertion Loss	Bandwidth	Driving Voltage V_π
EOSPACE Inc AX-CK5-10-PFU-SFU-UV	1550nm	<5dB	>10GHz	<3V@1GHz

From Table 2, the start index of the RSNS for shift $[0\ 1\ 2]^T$ is 733 and ends at 860. As shown in Table 4, the column vectors for position index 733 and 860 are $[6\ 4\ 7]^T$ and $[6\ 1\ 1]^T$, respectively. To obtain the integer values contained in the starting column vector, each interferometer transmissivity must be phase shifted by adding either a positive or negative DC voltage. The required DC bias voltages are positive 19, 13 and 31 LSBs (LSB = 102 mV) away, respectively.

The modulated optical signals are aligned to generate column vectors that start with $[6\ 4\ 7]^T$ using the same process outlined in Chapter II. The MATLAB generated RSNS and thermometer code for left shift $[0\ 1\ 2]^T$ at starting position index $h = 733$ with symmetrical residue $[6\ 4\ 7]^T$ are as shown in Figures 9, 10 and 11.

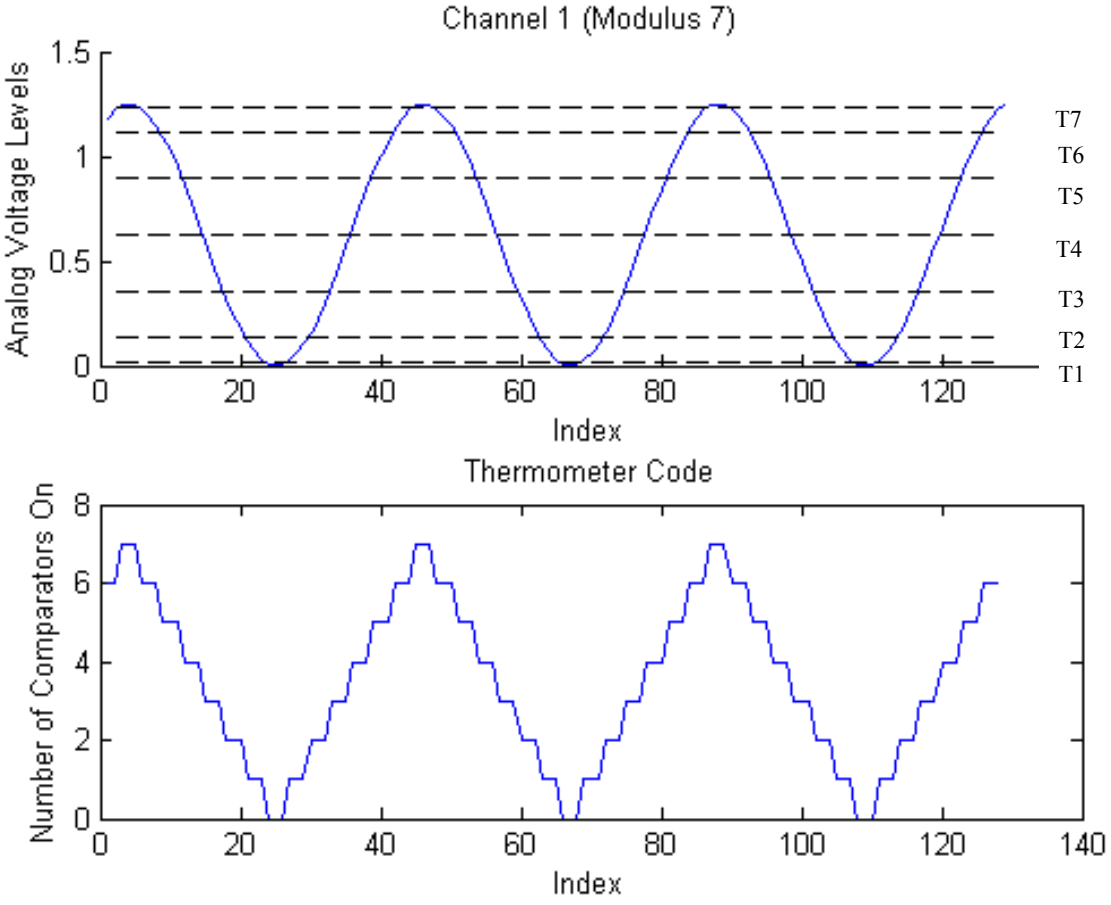

Figure 9. RSNS waveform and thermometer code, $(\mathrm{mod}\,7, s_{shift_1} = 0)$.

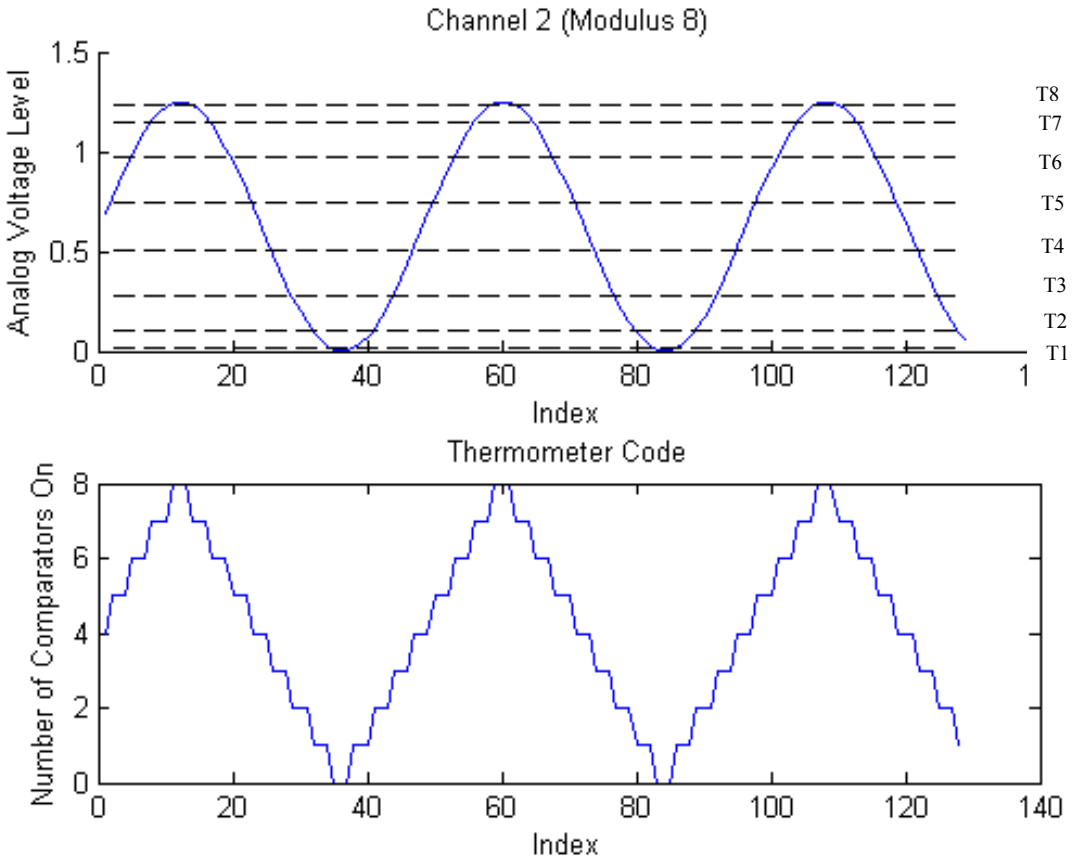

Figure 10. RSNS waveform and thermometer code, $(\mathrm{mod}\,8, s_{shift_2} = 1)$.

The threshold values are also shown for each channel. With the waveforms properly aligned, the combination of integers in the first fifteen column vectors are shown in Table 9. For example at h = 733, the amplitude of modulus 7 signal is greater than the threshold value T6. Thus, the integer value at that position is 6 (number of comparators on). With modulus 8, the amplitude is greater than T1 … T4. Hence, an integer value of 4 is also obtained. Lastly, the value of modulus 9 signal is greater than its first seven threshold values, which results in integer 7. The integer values represent the symmetrical values within each channel.

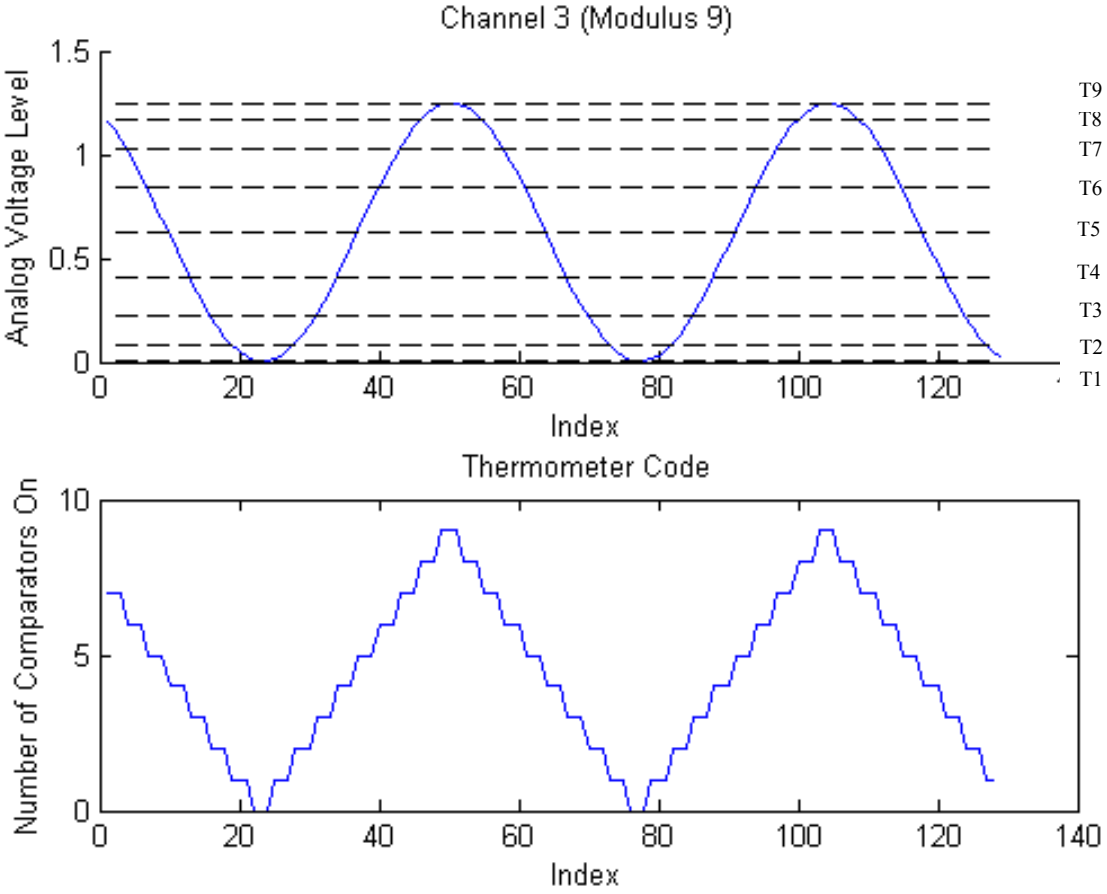

Figure 11. RSNS waveform and thermometer code, $(\mathrm{mod}\,9, s_{shift_3} = 2)$.

Observe that the interval between the threshold voltages becomes smaller as it approaches the maximum and minimum point of the folding waveform. The modulus-9 signal clearly demonstrates this characteristic, which is caused by the non-linearity of the folding waveform and higher fundamental period. Therefore, a careful examination of the threshold values must be made when using higher moduli values to prevent errors caused by added noise. When the threshold interval is less than the amplitude of the added noise, encoding errors occur.

Table 9. First fifteen column vectors.

mod 7	6	6	7	7	7	6	6	6	5	5	5	4	4	4	4	…
mod 8	4	5	5	5	6	6	6	7	7	7	8	8	8	7	7	…
mod 9	7	7	7	6	6	6	5	5	5	4	4	4	3	3	3	…
h	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	…

D. OPTICAL SIGNAL CONVERSION

After electro-optic modulation, the optical beam must be converted to an electrical signal prior to sampling. This can be done using a photodetector. The photodetector selected needs to have a large response at the wavelength to be detected, a small value for the noise introduced by the detector and sufficient speed of response to follow variations in the optical signal being detected.

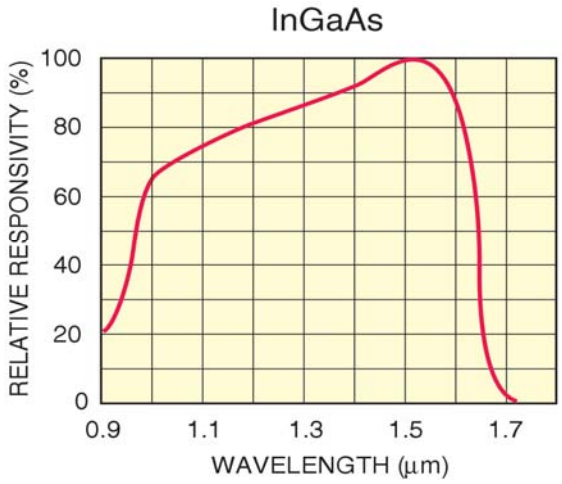

Figure 12. Relative responsivity of InGaAs based photodetector (From [37]).

To fulfill the above requirements, the key performance parameters to look out for in selecting a suitable photodetector are the operating wavelength, bandwidth, rise time, responsivity and noise equivalent power (NEP). The relative responsivity of photodetectors made from InGaAs is as shown in Figure 12. It can be seen that the responsivity is almost 100% for 1550 nm. The response time of the photodetector is characterized by the rise time. A short rise time will enable the detection of fast pulse. From the rise time, we can calculate the high frequency cutoff (equivalent to bandwidth), which is about 39 GHz, quite close to the manufacturer's specification of 45 GHz. The

amount of optical power incident on the surface of a photodetector that produces a signal at the output of the detector just equal to the noise generated internally by the detector is the NEP. A low NEP will allow a good signal-to-noise ratio (SNR), thereby minimizing the quantization errors. The performance parameter of the chosen photodetector is as shown in Table 10. The noise equivalent power (NEP) of the InGaAs photodetector is $< 45\ \text{pW}/\sqrt{\text{Hz}}$, which is very small. Thus, the noise coming from the photodetector is not the dominant noise source.

Table 10. Key performance parameters of the choice photodetector.

Model	Wavelength	Rise time	Bandwidth	Responsivity	Noise Equivalent Power
New Focus 1014	950-1650 nm	9 ps	45 GHz	0.4 A/W	$< 45\ \text{pW}/\sqrt{\text{Hz}}$

E. NOISE REDUCTION

Noise reduction is also important in the preprocessing. Using filters is one of the simplest methods to reduce the noise. A RC low-pass filter was used in the prototype [30]. Depending on the frequency of the signal of interest, the cutoff frequency can be adjusted using

$$f_c = \frac{1}{2\pi RC} \tag{3.5}$$

where R is the resistor value and C is the capacitance. The capacitance was developed by using several capacitors in parallel with the DC power supply to reduce the noise level prior to amplification.

F. SUMMARY

The signal processing steps to implement the RSNS preprocessing front-end architecture were described in this chapter, and the characteristics and considerations in the choice of hardware for the RF preprocessing circuit, optical modulation circuit and the optical to electrical conversion or detector circuit were explained. Finally, the chosen hardware was highlighted together with the key performance parameters. A summary of the components selected is shown in Table 11. In the next chapter, a detailed discussion of the results will be presented.

Table 11. Summary of selected components.

Component Type	Selected Components
Power Amplifier	Nextec-RF (NB00415)
Fixed Attenuator	Bird Technologies (100-A-MFA-03/06)
Variable Attenuator	Agilent (8494A Manual Step Attenuator)
Interferometer	EOspace (Model AX-CK5-10-PFU-SFU-UV Low Vpi 10 Gb/s X-cut Lithium Niobate Intensity Modulator at 1550nm Wavelength)
Photodetector	New Focus (45-GHz InGaAs Photodetector Model 1014)
Optics Splitter	Ozoptics (1×3 splitter FUSED-13-1550-8)
Power Supply	Instek (GPS-3030DD)

THIS PAGE INTENTIONALLY LEFT BLANK

IV. PREPROCESSING RESULTS AND DATA ANALYSIS

This chapter begins with the characterization of the hardware components. Limitations leading to design and implementation changes are discussed. The respective RSNS waveforms are then generated aligned with the new design. This is performed through application of the RF voltages and DC shift voltages for the individual RSNS channel.

A. COMPONENT CHARACTERIZATION

To ensure that the components meet the specifications, a series of experiments are conducted to verify the performance parameters of each component.

1. Power Amplifier

The specifications for the power amplifier includes being able to generate a gain of 40-50 dB within a bandwidth of 1-500 MHz. The amplifier output corresponding to three different inputs at 1 MHz is shown in Table 12. It can be seen that for a fixed frequency, the gain is constant for the three input values. The measured gain is within specifications. The measurement setup for the power amplifier and attenuators is as shown in Figure 13.

Table 12. Calculated gain of power amplifier at 1 MHz.

Input	S/N 0238		S/N 0239		S/N 0240	
	Output	Gain	Output	Gain	Output	Gain
10 mV/ -27 dBm	17 dBm	27+27 = 44 dB	19.5 dBm	27+19.5 = 46.5 dBm	22 dBm	27+22= 49 dBm
25 mV/ -19 dBm	25 dBm	19+25 = 44 dB	27.5 dBm	19+27.5 = 46.5 dBm	30 dBm	19+30= 49 dBm
30 mV/ -17.5 dBm	29 dBm	17.5+26.5 = 44 dB	29dBm	17.5+29 = 46.5 dBm	32 dBm	17.5+32= 49.5 dBm

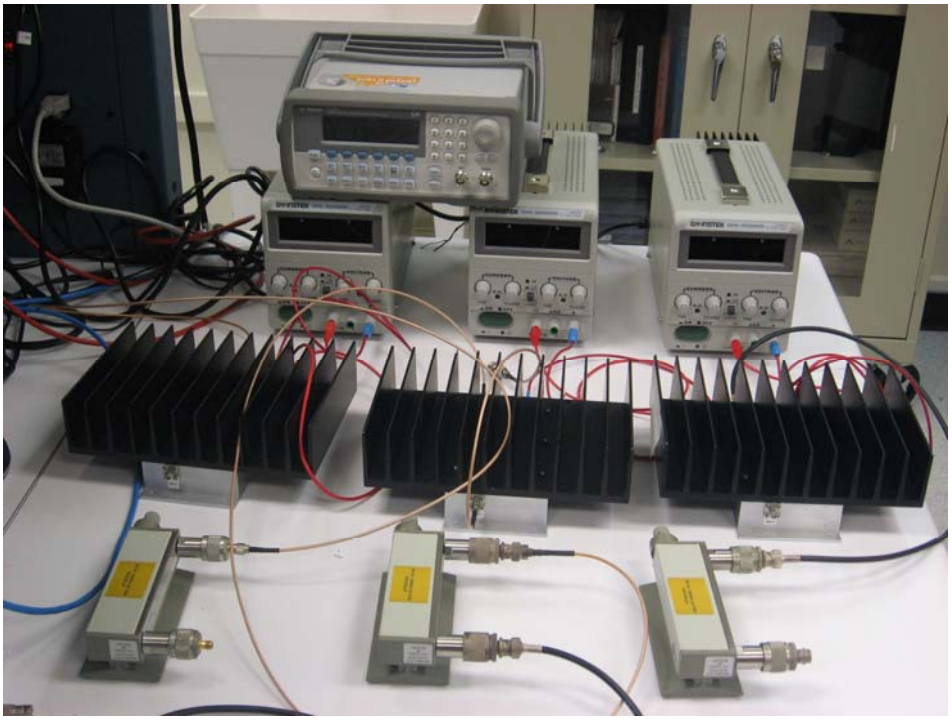

Figure 13. Measurement Setup.

2. Fixed and Variable Attenuators

A 1 MHz signal with a 10 mV peak-to-peak voltage was sent to the 3 dB and 6 dB attenuators. The output values are – 30 dBm and – 33 dBm, respectively, in line with the expected results. The measurement setup is similar to Figure 13.

3. Photodetectors

The photodetectors were connected with a 1550 nm laser source and the output was monitored on the oscilloscope. All three photodetectors were within specification, i.e., maximum conversion gain of 10 V/W.

4. Modulators

The modulators were tested with a 1550 nm laser source modulated with a 1 kHz RF signal. DC bias was applied via the DC port. All three modulators showed a V_{π} value <3 V for the RF bias port and <5.5 V for the DC bias port, in line with the specification.

B. SETUP CHANGES

The output from the three power amplifiers with a 1 MHz sine function is as shown in Figure 14. The output is generated with a function generator via a 1:3 RF splitter. With the power amplifier in place, the required RF input can be reduced to a minimum (in line with what is happening in the real world), thereby allowing more latitude in the power control of each channel. Due to limitation in the sampling circuit [31], the RF input signal is optimized to 1 kHz. For this setup to function, the input RF signal need to have a minimum of 1 MHz (Bandwidth of power amplifier 1-500 MHz). Though not implemented in the current setup, this proposed design can be applied to future setup once the limitation on the sampling circuit is overcome.

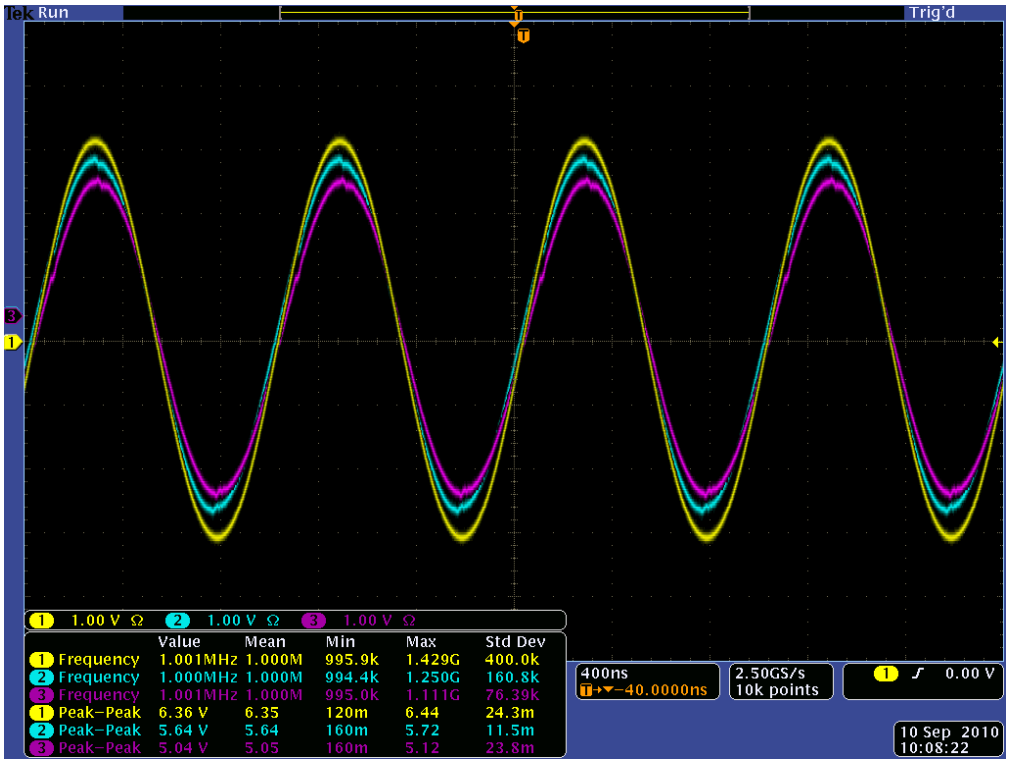

Figure 14. Synchronized RF output using power amplifier.

A new approach utilizing three synchronized function generators was proposed to provide the power required for each channel. This allows great flexibility in the power control for each channel. The key is to synchronize all three function generators to produce three coherent ramp functions such that they cross the (0,0) point at one-half the

maximum voltage. The setup of the three function generators is as shown in Figure 15. The digital function generator was selected as the clock source while the other two analog function generators were selected due to their higher power output level (15 V_{p-p}) as compared to the digital function generator (10 V_{p-p}). The synchronized ramp functions are as shown in Figure 16.

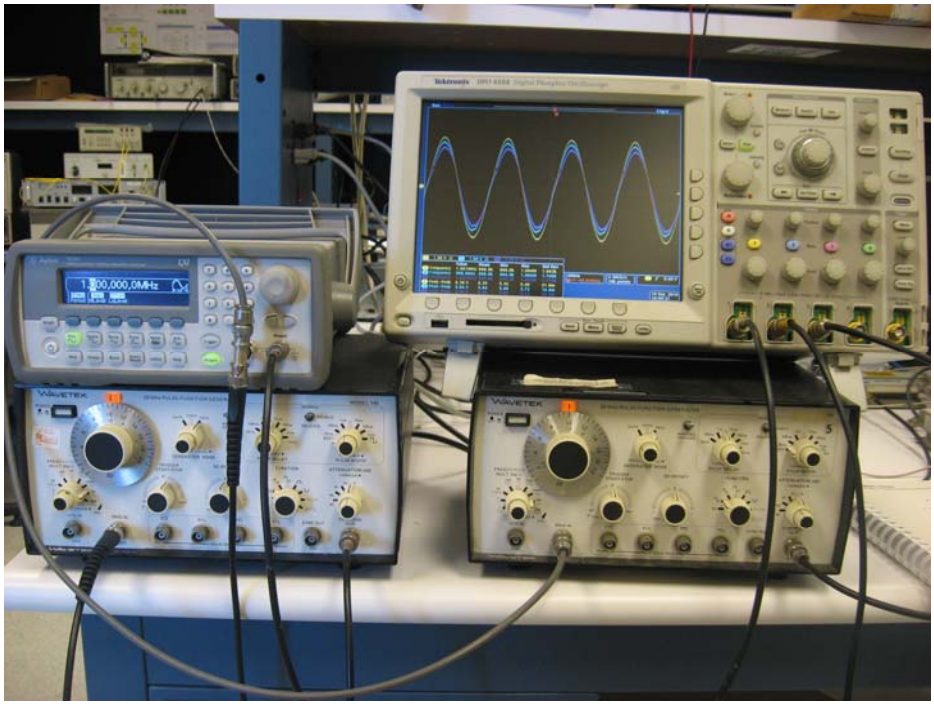

Figure 15. Function Generator Setup.

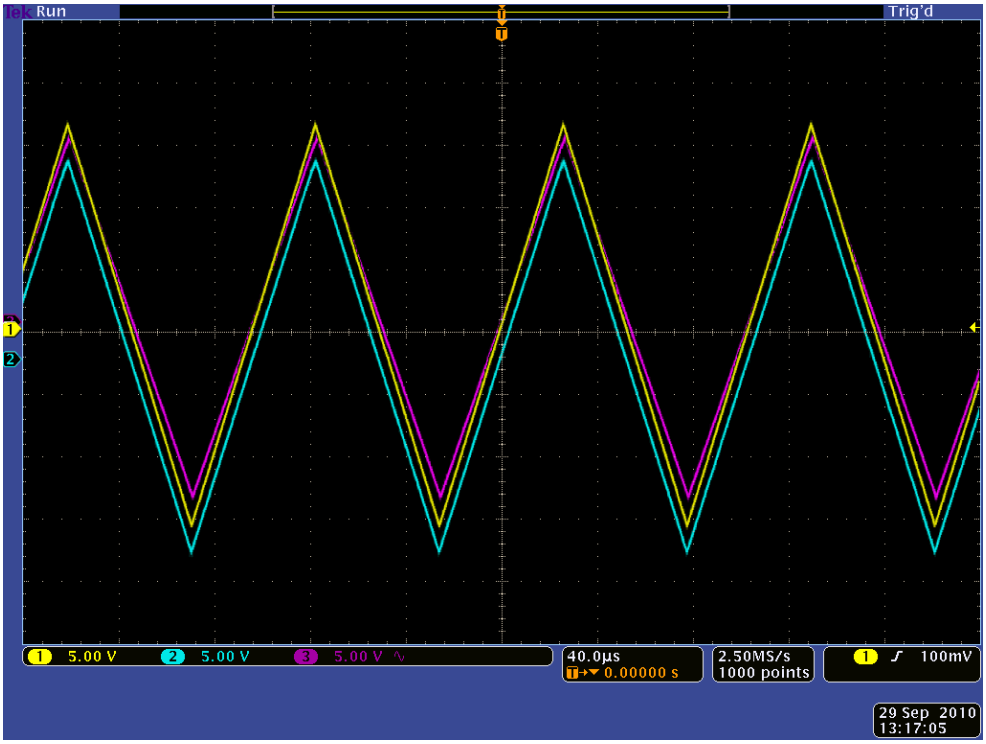

Figure 16. Coherent 25-kHz Ramp Functions.

C. EXPERIMENTAL SETUP AND RESULTS

The V_{RF_i} input to the three modulators to derive the number of folds required for each channel is as shown in Figures 17, 18 and 19. The V_π measurements are also performed for each modulator. The superimposed RSNS waveform for channels 1, 2 and 3 is as shown in Figure 20. The results are as shown in Table 13.

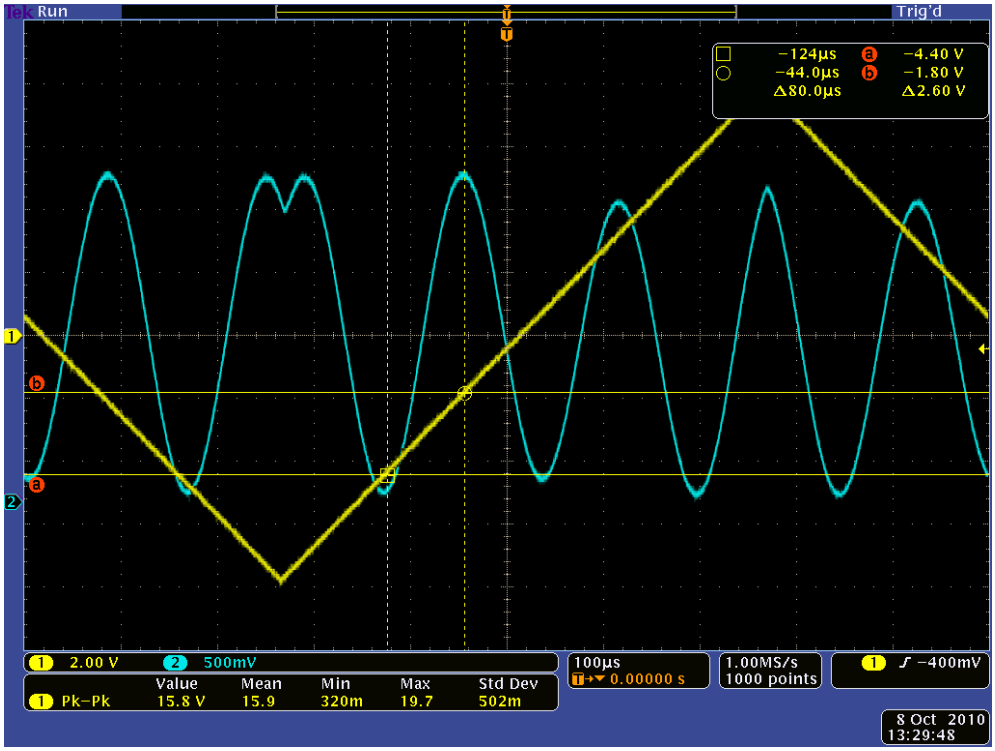

Figure 17. Aligned RSNS waveform for mod 7 (channel 1).

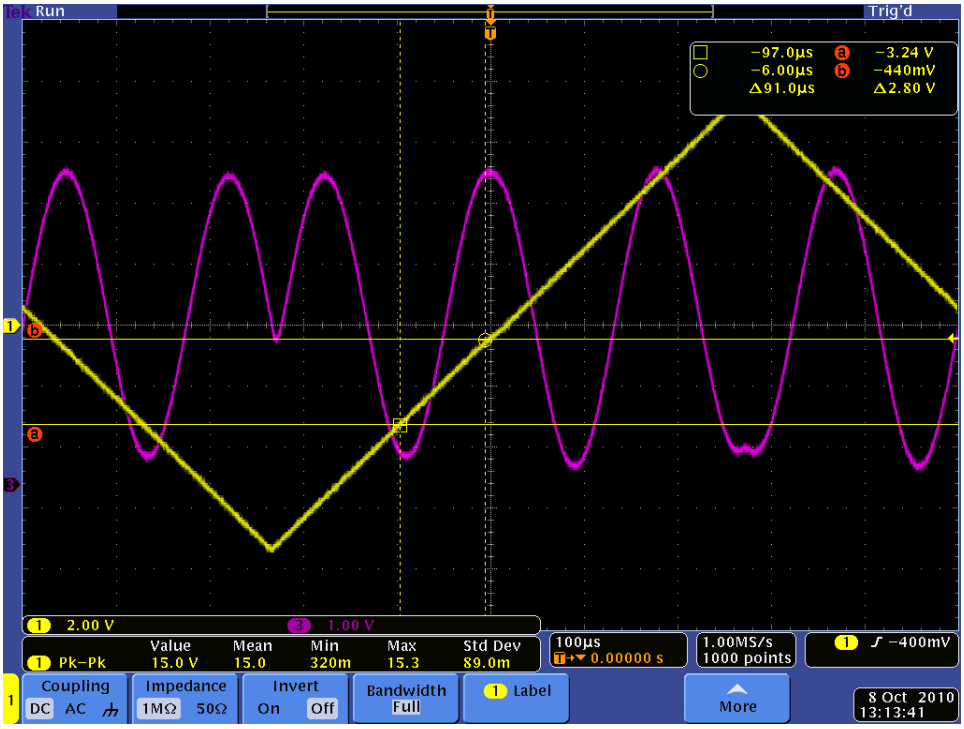

Figure 18. Aligned RSNS waveform for mod 8 (channel 2).

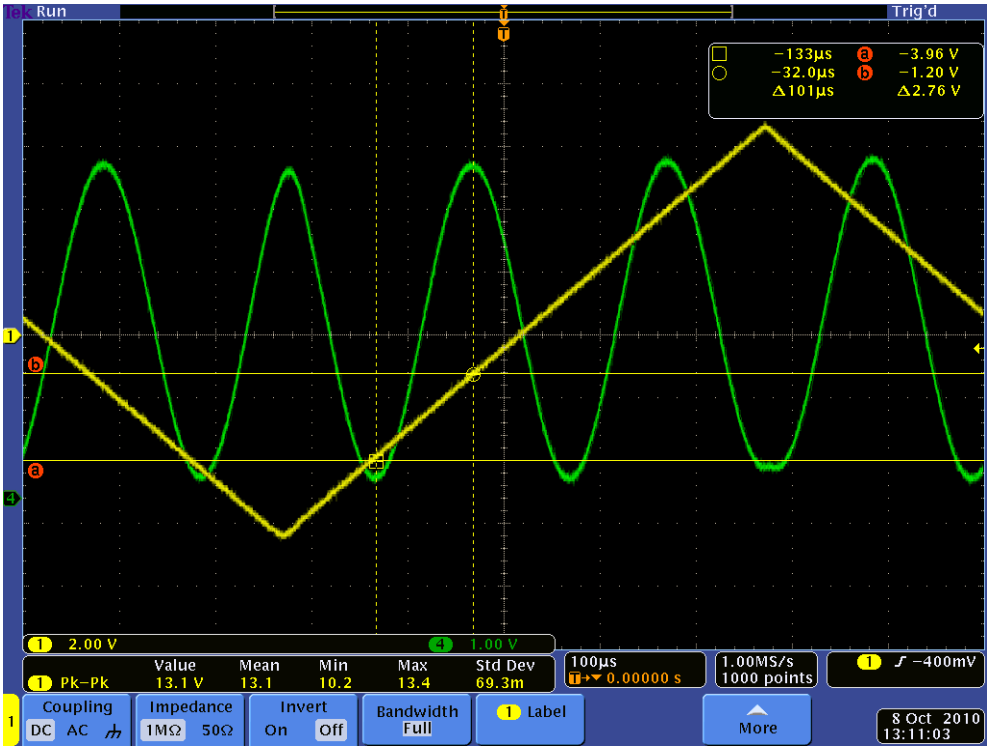

Figure 19. Aligned RSNS waveform for mod 9 (channel 3).

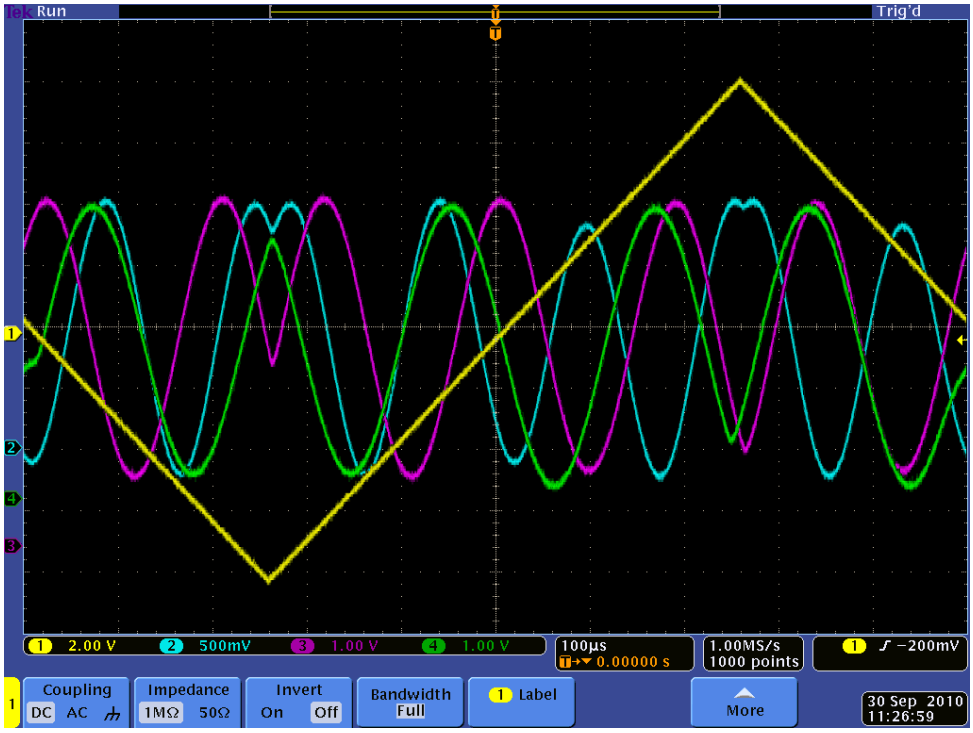

Figure 20. Aligned RSNS waveform for mod 7, 8 and 9.

Table 13. Experimental results and calculations based on 7-bit implementation.

Channel/ Modulator S/N	V_{RF_i} V Calc	V_{RF_i} V Meas	V_π V Meas	V_{DC_i} Calc	V_{DC_i} V Meas.	K_i	LSB mV Calc
1 203919	9.14	15.9	2.60	1.9	2	1.21	102
2 203922	8	15.0	2.80	1.3	1.4	1.14	102
3 203925	7.1	13.1	2.76	3.2	3.3	1	102

1. Changes in Dynamic Range.

From Table 12, the measured V_{RF_i} values are approximately double that of the calculated values. This is due to the different terminations (50 Ω termination was assumed for the calculations but the actual input termination was 1 MΩ) as a result of the differing RF port input impedances of the modulators. As can be seen from Figures 17, 18 and 19, the modulated waveforms exhibit periodic uneven amplitude output. When the aligned waveforms are sampled, the original input RF ramp function could not be recovered completely. This is due to the sampling errors introduced as a result of the uneven waveform. It is suspected that there may be an anomaly in the behavior of the material substrate of the modulators.[1] To overcome the constraint, an alignment was performed for the dynamic range of 41 instead of 128 using the current design, i.e., 6-bit implementation using a 7-bit design. The new dynamic range is obtained through the optimization of the areas of even modulation of all three channels. Alignment for the new dynamic range was achieved via the DC bias port which allowed phase control of each channel. The aligned waveforms are as shown in Figure 21. The results and calculations are as shown in Table 13. The recovered ramp function is as shown in Figure 22.

[1] Screen captures of the results were sent to the manufacturer and they have acknowledged that the output is not normal. Due to time constraint, the modulators will be sent back to the manufacturer for evaluation.

2. K_i value derivation and LSB calculation.

The K_i value for each channel is calculated based on the ratio of the V_{RF_i} applied to channels 1, 2 and 3 to the V_{RF_i} applied to channel 3. This is similar to assuming a single function generator with a 1:3 RF splitter applied with the necessary K_i value to generate the necessary V_{RF_i} required for each channel. Based on the above, the K_i value for channels 1, 2 and 3 are 1.21, 1.14 and 1, respectively. The LSBs are obtained using (2.8). It can be seen that the K_i and LSBs calculated are similar for all three channels in Table 13 and Table 14.

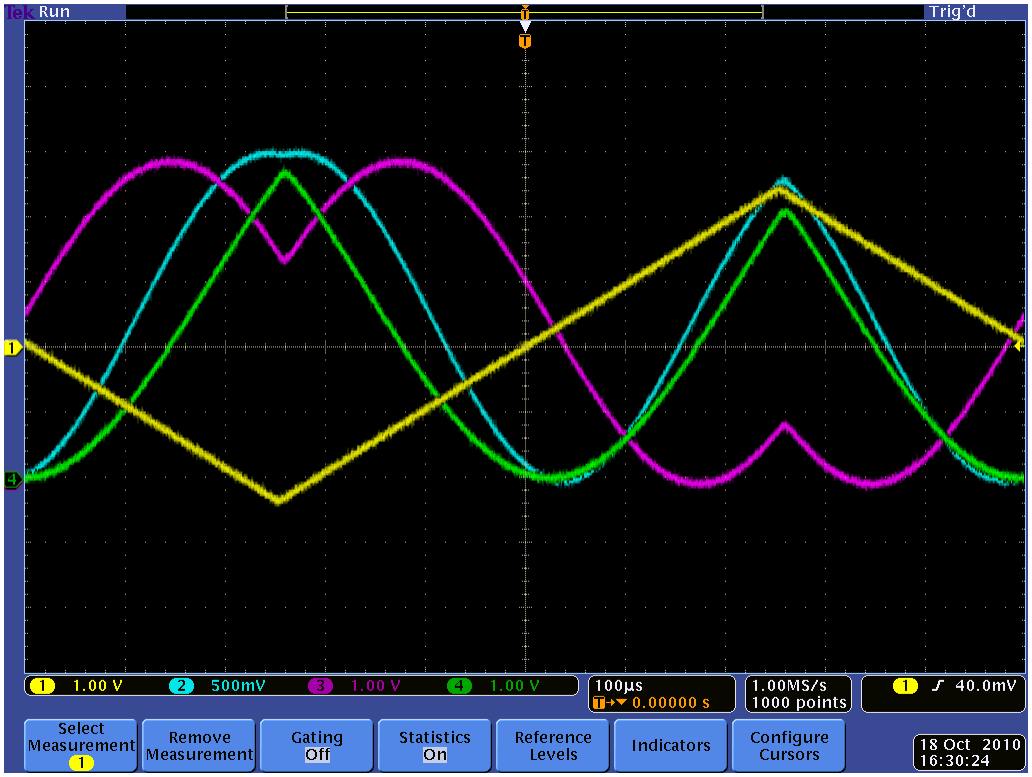

Figure 21. Aligned waveforms based on dynamic range of 41.

Table 14. Results and calculations based on 6-bit implementation.

Channel/ Modulator S/N	V_{RF_i} V Meas	V_π V Meas	K_i	LSB mV Calc
1 203919	4.70	2.60	1.22	102
2 203922	4.45	2.80	1.15	102
3 203925	3.86	2.76	1	102

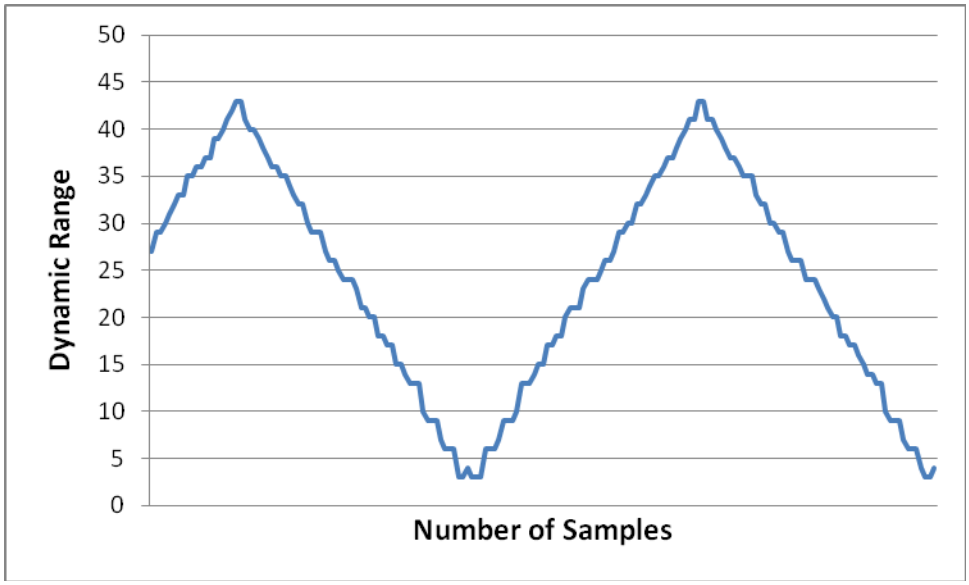

Figure 22. Recovered ramp function.

D. KEY OBSERVATIONS

1. Signal alignment process

The importance of the input RF voltage needed to generate the correct number of folds and the DC bias control for proper alignment of the signal cannot be over-emphasized. For a more efficient and effective alignment process, the following steps are recommended:

a) Use three synchronized digital function generators to supply the required V_{RF_i} to each modulator to generate the required number of folds,

b) Align the three outputs from the modulators based on the MATLAB output and calculate V_{DC_i}. Keep one of the channel values constant, i.e., minimum modulus channel while tuning the other channels,

c) Ensure that all three waveforms are aligned to the same DC offset on the scope to ensure consistent sampling by the comparators,

d) Adjust up the amplitude of the aligned signals using the post-processing circuit. This allows for a more robust sampled signal output since the amplitude of the signal is a function of the threshold values calculated for the comparators.

E. SUMMARY

The characterization of the hardware components prior to the build-up of the preprocessing architecture were described in this chapter. The limitations leading to design changes and the results were highlighted and discussed at length. Lastly, key observations and lessons learnt are highlighted and shared for a more efficient and effective alignment process. In the next chapter, the dynamic performance of the ADC will be presented.

THIS PAGE INTENTIONALLY LEFT BLANK

V. DYNAMIC PERFORMANCE ANALYSIS

In this chapter, a number of important parameters that describe a converter's performance are presented. Differential and integral linearity errors are plotted for the ADC to analyze the linearity errors. The dynamic range is determined using a full-scale sinusoid and a subsequent Fourier spectrum analysis of the noise floor. Finally, the signal's spectrum is characterized by the SNR, SNR plus distortion (SINAD) and the effective number of bits (ENOB).

A. LINEARITY ERRORS

The characteristic transfer function of a 1 kHz triangular waveform is used to determine the linearity of the ADC. The input voltage and the decimal output voltage plot are as shown in Figure 23. A quantization error is present as there is not a one-to-one correspondence between the input and output voltage values. To quantify the linearity error, the step size is first computed [35]. The differential nonlinearity (DNL) is then

$$DNL_k = V_k - V_{k-1} - LSB \tag{5.1}$$

where V_k, V_{k-1} are two consecutive code transition points and $V_k - V_{k-1}$ is the step size. The DNL is the maximum deviation in the output step size from the ideal value of one LSB. The integral nonlinearity (INL) is then [35]

$$INL_j = \sum_{k=1}^{j} DNL_k = V_j - jLSB \tag{5.2}$$

where $V_j = \sum_{k=1}^{j} (V_k - V_{k-1})$ is the sum of the step size from zero to the j^{th} transition point and $jLSB$ is the ideal value at that transition point. The INL is the maximum deviation of the input/output characteristic from a straight line passed through its end points. A good converter typically has linearity error ≤ 0.5 LSB [35].

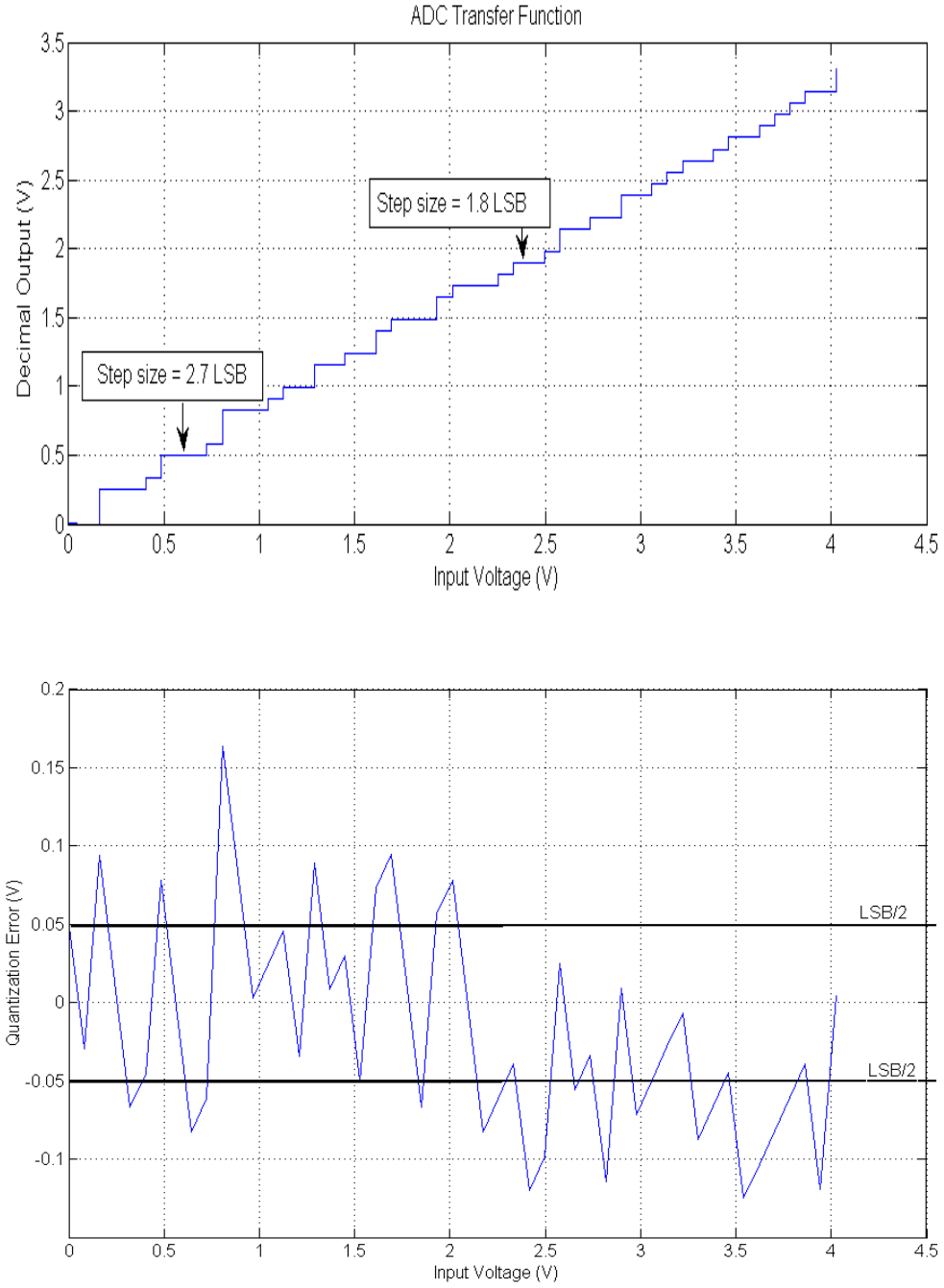

Figure 23. (a) Transfer function of photonic ADC using a 1 kHz triangular wave as input signal, (b) Quantization error.

The linearity errors of the quantized signal in Figure 23 are examined in Figure 24. The step size plot depicts the length of input voltage corresponding to each quantization level as the input voltage increases. From the step size plot, we see that the maximum step size is 2.7 LSB. The plot also shows the DNL and INL with a maximum INL value of 1.7 and 7.8 LSB, respectively.

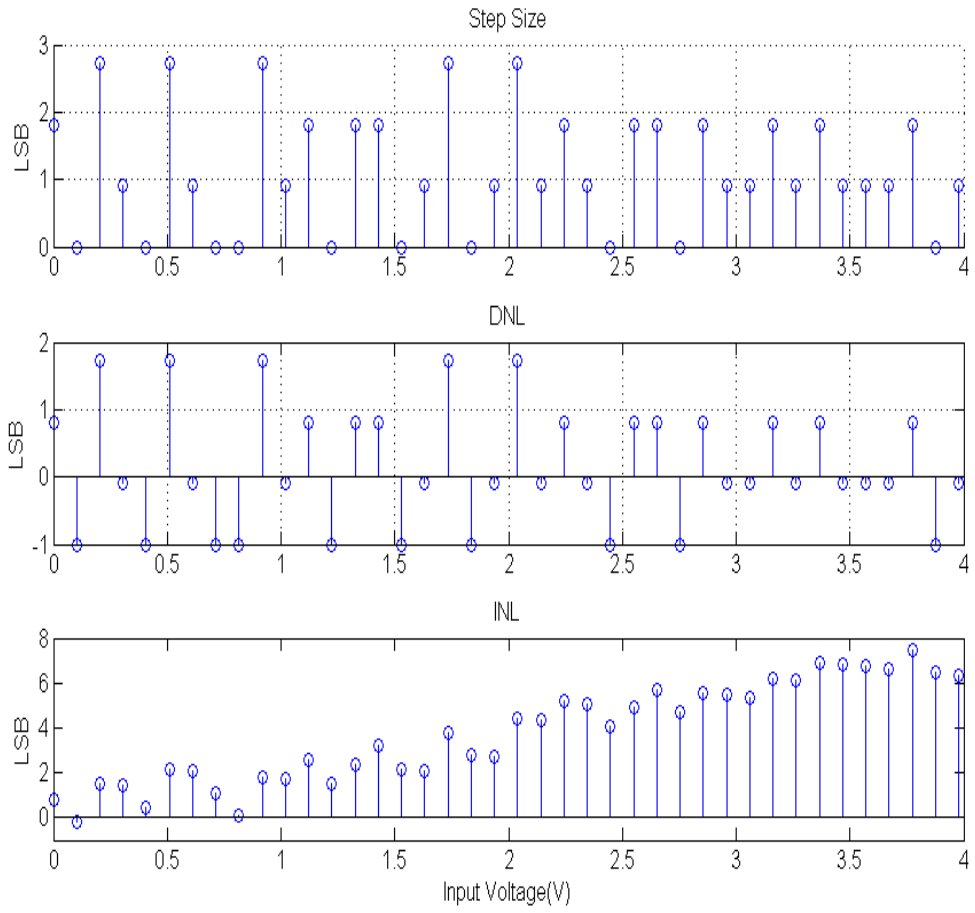

Figure 24. Linearity parameters showing the step size, differential nonlinearity and integral nonlinearity.

B. NOISE FLOOR ANALYSIS

To determine the noise floor of the ADC, a full-scale sinusoid is used as an input. In this section, the noise floor of the ADC is analyzed using a Blackman-Harris window. Twenty sets of 4,096 digitized samples were gathered. By performing spectral averaging,

the average magnitude spectrum response was obtained. A sinusoidal signal is processed by the ADC and a white noise process (quantization noise) γ_k is added to the digitized signal. The window samples are represented by w_k. Using the discrete Fourier transform (DFT), we obtained the point-by-point spectral average [35]. The process for computing the frequency spectrum and analyzing the noise floor is as illustrated in Figure 25.

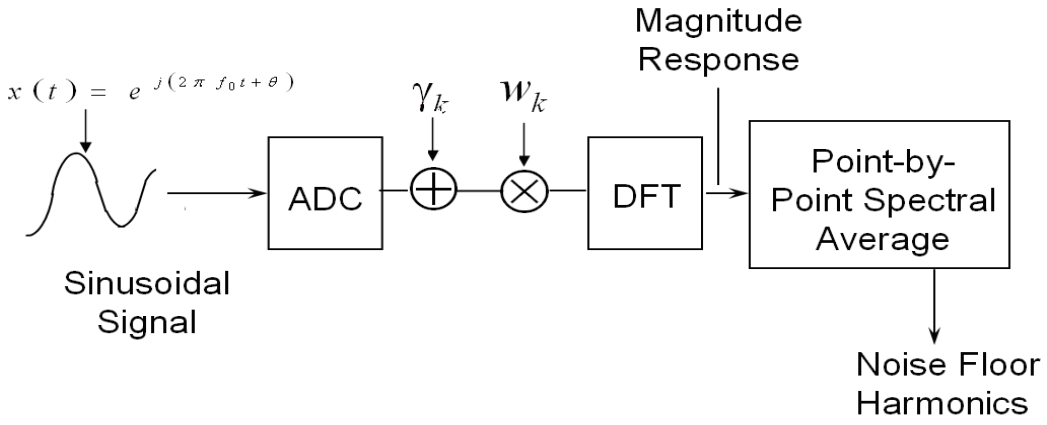

Figure 25. Process for examining the noise floor of the photonic ADC (From [35]).

To perform spectral analysis and noise floor examination, a sinusoidal signal was sampled at two different frequencies. Twenty sets of digitized signals were acquired, each containing 4,096 samples for each frequency. A Blackman-Harris window was used to window the collected samples because of the low sidelobe levels achieved [35]. To compute the spectrum of the signal, the fast Fourier transform (FFT) function in MATLAB was used. After computing the transform of each signal, the point-by-point spectral average was calculated. The spectral average of a 1- and 2- kHz sinusoidal signal with a Blackman-Harris window and N = 4096 is as shown in Figures 26 and 27. The noise floor levels are – 64.5 dB and – 59.5 dB, respectively.

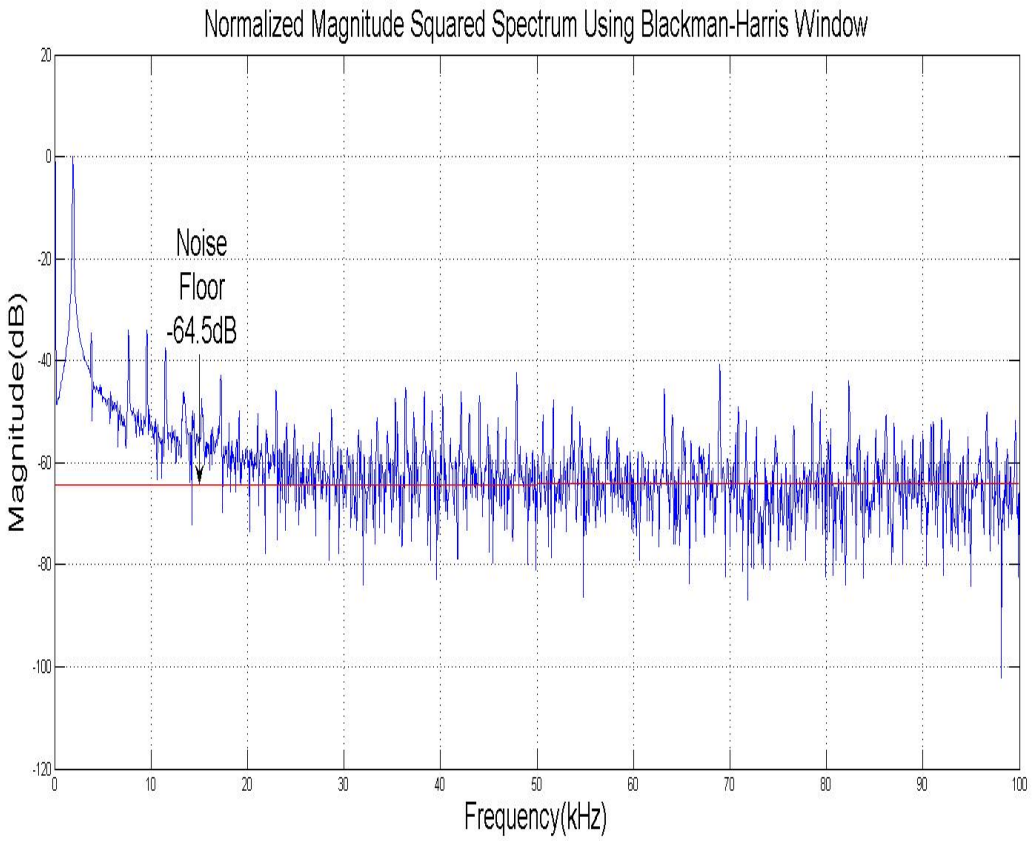

Figure 26. Spectral average of a 1 kHz sinusoidal signal with a Blackman-Harris window and $N = 4096$.

The noise floor is evaluated by examining the signal-to-noise ratio (SNR) and considering the presence of quantization noise only. By normalizing the magnitude square spectral average with the fundamental signal, the equation to calculate the noise floor using a Blackman-Harris window is [35]

$$F_{M2} = 10\log_{10}\left(\frac{3M}{4E_B}\right) + 6.02n \text{ dB} \tag{5.3}$$

where $n = \log_2(48) = 5.58$ bits and $E_B = 2.0$ is the equivalent noise bandwidth of the Blackman-Harris window. From this expression, the theoretical noise floor is – 65.49 dB. The measured noise floor from Figure 26 is – 65.5 dB. Since the noise floor is similar to the theoretical noise floor, the systems other additive noise sources (such as thermal noise) are less significant as compared to quantization noise.

1. Clock Jitter

Two test frequencies were separately used as input signals to test if clock jitter is a dominant noise source [35]. A 2 kHz signal was sampled to compare its noise floor with the 2 kHz sinusoidal signal. The same process in calculating the magnitude square spectrum of the 1 kHz signal was used for the 2 kHz signal.

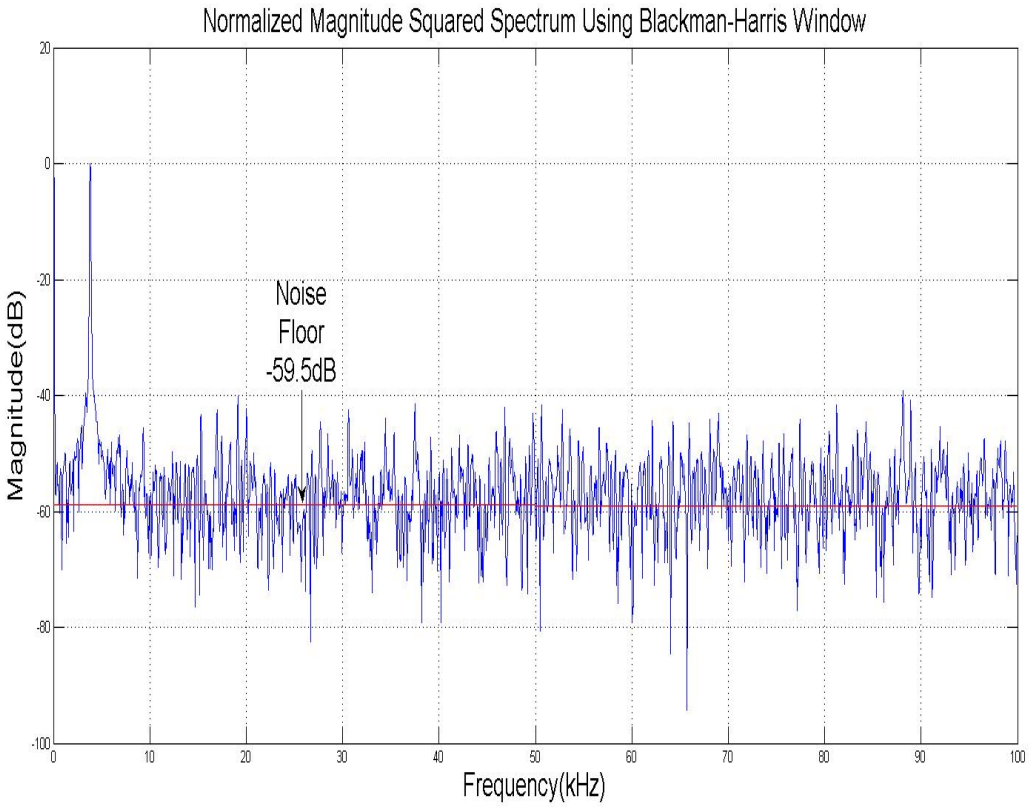

Figure 27. Spectral average of the 2 kHz sinusoidal signal using a Blackman-Harris window and N = 4096.

The normalized spectral average of a 2 kHz signal using Blackman-Harris window is shown in Figure 27. Comparison of the noise floor of both the 1 kHz and 2 kHz signals shows that the difference is within 6 dB. Thus, the clock jitter is not a dominant noise source [35], which is reasonable since this is a low frequency test signal.

C. FOURIER SPECTRUM ANALYSIS

The signal's spectrum is examined using the SNR, THD, SINAD and ENOB. Two different frequencies were digitized one at a time, starting with 1 kHz, followed by 2 kHz.

The ideal SNR equation is [35]

$$SNR(dB) = 6.02n + 1.76 \tag{5.4}$$

where n is the number of bits. With $n = \log_2 \hat{M}$, the calculated ideal SNR is 34.01 dB.

The THD expression used in MATLAB calculation is [35]

$$THD(dB) = 20\log\sqrt{\left(10^{2^{nd}\ HAR/20}\right)^2 + \left(10^{3^{rd}\ HAR/20}\right)^2 + \ldots} \tag{5.5}$$

where the input signal's first five harmonics are added and are in decibels. Using the SNR and THD values to express the SINAD equation, we get [35]

$$SINAD(dB) = -20\log\sqrt{10^{-(SNR)/10} + 10^{THD/10}}. \tag{5.6}$$

After calculating the SINAD, we can express ENOB as [35],

$$ENOB = \frac{SINAD - 1.76 + 20\log\left(\dfrac{\text{full scale amplitude}}{\text{actual input amplitude}}\right)}{6.02}. \tag{5.7}$$

The calculated ADC parameter values in dB for each input frequency are as shown in Table 15. With the 2 kHz signal, the ENOB is 0.0714 bit less than that for a 1 kHz signal. The THD value for the 2 kHz signal is 5.9922 dB higher than that for the 1 kHz signal. The ENOB and SINAD values decrease as the frequency of the sampled signal increases due to signal distortion. A comparison of the ADC 1 kHz input signal and the FPGA output signal is as shown in Figure 28. It shows that the FPGA output signal is able to follow the input signal, with some quantization errors.

Table 15. Dynamic performance parameters of the ADC.

		Blackman-Harris Window		
Frequency	Ideal SNR	SINAD	THD	ENOB
1-kHz	34.0125	34.4199	-62.8375	5.4247
2-kHz	34.0125	33.9899	-56.8453	5.3533

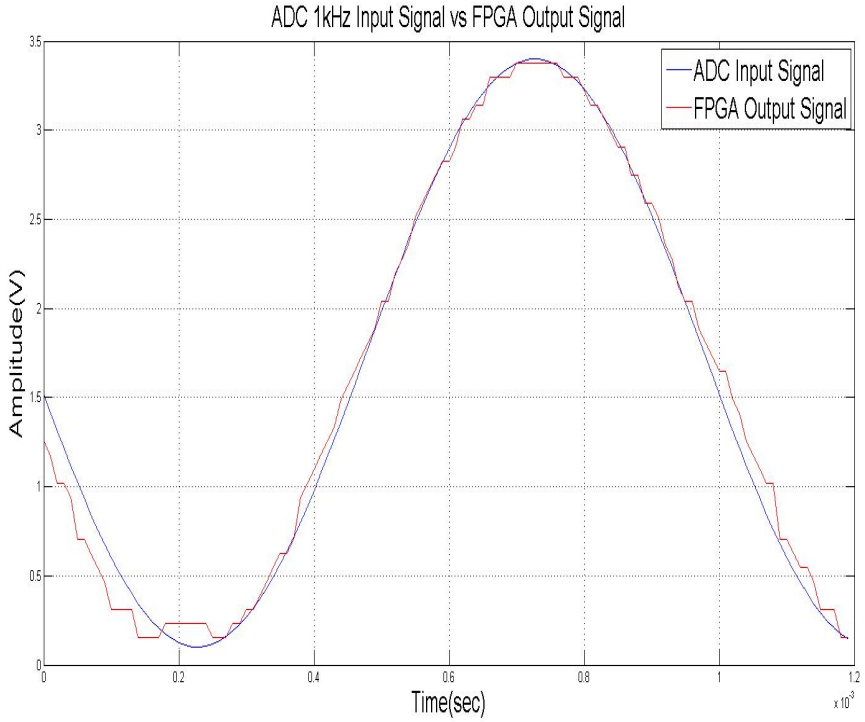

Figure 28. Comparison of ADC input signal and FPGA output signal.

D. SUMMARY

The performance of the ADC was described in this chapter. Firstly, the linearity of the transfer function of the ADC was determined. Secondly, the computed spectral average was analyzed and the noise floor was determined. Lastly, the calculated dynamic performance parameters of the ADC using the computed spectral average were presented. In the next chapter, the conclusion and recommendations will be presented.

VI. CONCLUSION AND RECOMMENDATIONS

The concept of an electro-optic ADC using the RSNS was explained and reiterated in this thesis. The RF front-end preprocessing architecture was designed and fully implemented by combining photonic and electronic devices. Due to uneven amplitude output observed from the output of the modulators, a 6-bit (dynamic range 41) ADC utilizing a 7-bit design was fully implemented. Analysis of results again [30] proves the viability of the electro-optic ADC preprocessing architecture. During the integration stage, we analyzed electro-optic modulation and signal alignment. The FPGA encoding scheme was also examined in a separate thesis [31].

A. CONCLUSIONS

The design and specifications of the individual hardware components required for the 7-bit electro-optic ADC were discussed in Chapter III. In the design phase, the importance of communication between the hardware and software implementation to understand each other's capabilities and limitations can never be over emphasized. The importance of accurate specification, sourcing and procuring the required hardware was learned. In the implementation phase, it is critical to become familiar with the laboratory equipment prior to connecting the hardware components. The alignment process proved less challenging than anticipated with an accurate MATLAB simulation to generate the aligned waveform, the availability of an individual power supply (i.e., three synchronized function generators) to adjust the required RF port input, plus the availability of an individual DC bias port to accurately align the phase of the waveforms. Due to uneven amplitude output observed from the modulators, a 6-bit (dynamic range 41) ADC utilizing a 7-bit design was fully implemented. Although the encoding scheme had minimal errors with low-frequency input signal, the errors increased as the modulated signals became distorted at higher frequencies. Signal alignment was also critical in obtaining an accurate encoding scheme.

The integration process also examined the FPGA encoding scheme [31]. After obtaining the accurately folded and aligned signals, the correct threshold values obtained

from simulation were programmed into the FPGA. This was essential in achieving the correct sequence of integers in the RSNS row vectors. In order to prevent errors during sampling, the FPGA internal clock synchronized the FPGA and the comparators [31].

Though the full implementation (i.e., 7-bit electro-optic ADC) was not achieved, a 6-bit ADC utilizing a 7-bit design was fully implemented. This proves the feasibility of the 7-bit implementation. With the uneven amplitude output from the modulators resolved, the 7-bit implementation can be realized. This thesis demonstrated the importance of having a separate RF and DC port input during the alignment process. With the flexibility of a separate power control, the waveform generation and alignment process was made much easier and more efficient. With further upgrades, the system can be used to digitize ultra-high bandwidth RF signals without the need for down conversion.

B. FURTHER IMPROVEMENTS AND STUDIES

The implementation of the ADC was limited to 1 kHz due to the unavailability of the hardware of the sampling circuit [31]. To harness the full potential of the implementation (at least 1 MHz), 1) procure a function generator that is able to generate a stable ramp function of at least 1 MHz, 2) replace the modulators with ones that are optimized for multi-half wave voltage applications and, 3) improve the sampling frequency of the postprocessing circuit by replacing the input module [31].

Further studies can also be focused on the effect of noise on smaller threshold intervals near the maximum and minimum of the folding waveforms.

Key to an upgrade in the electro-optic ADC design is the availability of a low V_π modulator. It is the single key enabler for a low power implementation in the RF front-end hardware implementation. The current low V_π modulators on the market (utilized in this thesis) have a value of 2.6-2.8 V. For an upgrade of the project to be feasible, the half- wave voltage values must be reduced further.

APPENDIX A. MATLAB CODE TO DETERMINE START AND END POINT FOR A RIGHT SHIFT SYSTEM

```
% THIS PROGRAM FINDS MAXIMAL STRINGS OF NON-REDUNDANT
% VECTORS FOR THE N (N >2 AND ANY INTEGER) CHANNEL ROBUST SNS FOR A
RIGHT-SHIFT SYSTEM.
% WRITTEN BY D. STYER and P. E. PACE

clear all

% DEFINE THE CHANNEL NUMBER,THE VALUES OF CHANNELS AND
% THE NUMBER OF SEARCH WRT FUNDEMENTAL PERIOD

disp('This program finds the maximal strings of non-redundant
vectors');
disp('for the N channel ROBUST SNS');
chanum=input('Enter the Number of Channels for ROBUST SNS >> ');

M=1;
for i=1:chanum
    m(i)=input(['Enter ' int2str(i) '.Channel Value >> ']);
    M=M*m(i);
end
period=2*chanum*M;
nsearch=period+10;

prompt='y';

% DEFINE THE SHIFT AMOUNT OF CHANNELS
while (prompt=='y')|(prompt=='Y')

    for i=1:chanum
        s(i)=input(['Enter ' int2str(i) '. Channel Shift Value >> ']);
    end

    % INITIATE THE VARIABLES TO ZERO
    i=0;
    ii=0;
    j=0;
    jj=0;
    k=0;
    funper=0;
    dynrange=0;

    % DEFINE THE WAVEFORM FOR CHANNEL m(1),m(2)...m(n) IN THE FORM OF
MATRIX g

    for r=1:chanum
        mm(r,[1 2])=[m(r) s(r)];

        for i=1+s(r):chanum*m(r)+s(r)
```

```
            g(r,i)=floor((i-s(r))/chanum);
        end

        for i=chanum*m(r)+1+s(r): 2*chanum*m(r)+s(r)
            g(r,i)=floor((2*chanum*m(r)+chanum-i+s(r)-1)/chanum);
        end

        g(r,2*chanum*m(r)+s(r)+1:4*chanum*m(r)+s(r))=...
            g(r,1+s(r):2*chanum*m(r)+s(r));
        g(r,4*chanum*m(r)+s(r)+1:8*chanum*m(r)+s(r))=...
            g(r,1+s(r):4*chanum*m(r)+s(r));
        g(r,8*chanum*m(r)+s(r)+1:16*chanum*m(r)+s(r))=...
            g(r,1+s(r):8*chanum*m(r)+s(r));
        g(r,16*chanum*m(r)+s(r)+1:32*chanum*m(r)+s(r))=...
            g(r,1+s(r):16*chanum*m(r)+s(r));
        g(r,32*chanum*m(r)+s(r)+1:64*chanum*m(r)+s(r))=...
            g(r,1+s(r):32*chanum*m(r)+s(r));
        g(r,64*chanum*m(r)+s(r)+1:128*chanum*m(r)+s(r))=...
            g(r,1+s(r):64*chanum*m(r)+s(r));
        g(r,128*chanum*m(r)+s(r)+1:256*chanum*m(r)+s(r))=...
            g(r,1+s(r):128*chanum*m(r)+s(r));
        g(r,256*chanum*m(r)+s(r)+1:512*chanum*m(r)+s(r))=...
            g(r,1+s(r):256*chanum*m(r)+s(r));
    end

    % DEFINE MATRIX ga AND THIS MATRIX IS THE TRANSPOSE OF MATRIX g

    ga=g';

    % DEFINE MATRIX gb AND THIS MATRIX GIVES THE ROW NUMBER IN COLUMN
1,
    % AND THE VECTOR FOR THAT ROW IN COLUMNS 2 THROUGH THE NUMBER OF
CHANNEL
    % DEFINED IN MATRIX ga

    gb(:,[2:(chanum+1)])=ga(:,[1:chanum]);
    [sgbr,sgbc]=size(gb);
    gb(:,1)=(1:1:sgbr)';

    % DEFINE THE MATRIX gc.THIS MATRIX GIVES THE ROW NUMBER THROUGH
    % THE NUMBER OF SEARCH (eliminate the parts of the matrix beyond
    % the search length)

    gc=gb;
    gc(nsearch+1:sgbr,:)=[];
    [sgcr,sgcc]=size(gc);

    % FIND THE FIRST REDUNDANCIES IN MATRIX gc

    k=1;
    for ii=2:nsearch; % ii is row index into gc
        xrec=gc(ii,[2:(chanum+1)]);
        for jj=ii+1:sgcr;
```

```
            if gc(jj,[2:(chanum+1)])==xrec
                redun=gc(jj,1);

                % DEFINE THE MATRIX h WHICH IS THE MATRIX OF
                % FIRST REDUNDANCIES

                h(k,1)=ii;
                h(k,2)=redun;
                k=k+1;
                break
            end
        end
    end

    % DEFINE THE MATRIX hsort AND SORT BY THE REDUNDANCY COLUMN IN
MATRIX h

    hsort=h;
    [yoy,ioi]=sort(hsort);

    % DEFINE THE MATRIX hsorted

    hsorted=[yoy(ioi(:,2),1) yoy(:,2)];
    hsorted;

    % DEFINE THE MATRIX hreduced.
    % ELIMINATE THE ROWS OF THE MATRIX hsorted THAT DO NOT ALLOW
    % THE FIRST COLUMN TO BE MONOTONE INCREASING
    % ssr - rows of hsorted
    % ssc - columns of hsorted
    % a - value in last row of h (hsort)
    % rx - rows in hreduced
    % cx - columns in hreduced

    [ssr,ssc]=size(hsorted);
    hreduced=hsorted;
    a=hsort(ssr,1);
    [rx cx]=size(hreduced);
    for k=1:ssr
        for i=1:ssr
            if i<rx
                if hreduced(i,1)==a
                    hreduced(i+1:rx,:)=[];
                    break
                elseif hreduced(i+1,1)<hreduced(i,1)
                    hreduced(i+1,:)=[];
                    break
                end
            end
            [rx cx]=size(hreduced);
        end
    end
    hreduced;
```

```
    % DEFINE THE MATRIX H THAT SHOWS WHICH SETS OF ROWS
    % ARE MAXIMAL FOR NO REDUNDANCIES AND THEIR LENGTHS.

    [hsr,hsc]=size(hreduced);
    H(1,1)=(chanum-1);
    H(2:hsr+1,1)=hreduced(1:hsr,1)+1;
    H(1:hsr,2)=hreduced(1:hsr,2)-1;
    H(hsr+1,2)=nsearch;
    H(1:hsr+1,3)=H(1:hsr+1,2)-H(1:hsr+1,1)+1;

    % FIND THE DYNAMIC RANGE OF N-CHANNEL RSNS

    HH=max(H);
    dynrange=HH(:,3);

    % DISPLAY A MATRIX THAT SHOWS THE BEGIN-END POSITION,
    % DYNAMIC RANGE AND ALSO CHANNEL-SHIFT VALUES

    disp(' ')
    disp(' ')
    disp(['THE FUNDEMANTAL PERIOD IS ',num2str(period),' '])
    disp(' ')
    disp(' ')
    disp(['THE DYNAMIC RANGE IS ',num2str(dynrange),' '])

    fprintf('\n    BEGIN POSITION  END POSITION  DYNAMIC RANGE\n')
    fprintf('    --------------  ------------  -------------\n')
    fprintf('%11.0f %16.0f %12.0f \n',H')


    fprintf('\n    CHANNEL VALUES     SHIFT VALUES\n')
    fprintf('    --------------     ------------\n')
    fprintf('%11.0f %17.0f \n',mm')


    prompt=input('Would you like to try another shift (y/n) ? >>','s');

end
```

APPENDIX B. MATLAB CODE TO DETERMINE START AND END POINT FOR A LEFT SHIFT SYSTEM

```
function [dynamic_range] = DynamicRangeSmartSearch(modli)

%*********************************************************************
****
%* DynamicRangeSmartSearch  Calculates the dynamic ranges for input set
of
%*     moduli. dynamic_range = DynamicRangeSmartSearch(moduli) returns
%*     a matrix whose rows contain information about the of all dynamic
%*     ranges for the input set of moduli. The columns of the matrix
%*     are as follows: Moduli Sum, Moduli (N of them), Fundamental
Period,
%*     DR lower bound, DR lower bound Case, DR upper bound, DR upper
%*     bound Case, and Dynamic Range (DR).
%*
%*     Author:         LCDR Brian Luke
%*     Last Modified: 31OCT2008 - added ArcGIS shapefile generation
%*
%*     Called Functions: crt, CalculateRedundancies,
Generate_RSNSCircle_Shapefile
%*                       Generate_CircleArc_Shapefile
%*     Calling Functions: startRSNSsearch
%*
%*********************************************************************
****

global CMIN CMAX N Pf MODLIST COUNT COMBINATIONS MIN_DYNRANGE
MAX_DYNRANGE RADIUS
global chan

format compact

N = length(modli);

dyn_ranges = [];
comb_dyn_ranges = [];
intervals = [];

% Fundamental period for PRP moduli
Pf = 2*N*prod(modli);

% Set the radius of the RSNS circle
RADIUS = round(Pf/(2*pi));

% generate RSNS circle plot filename and shapefile
fname = [num2str(N),'Channel'];
for i = 1:N
    fname = [fname,'_',num2str(modli(i))];
end
```

```
fname = [fname,'_CirclePlot'];
%Generate_RSNSCircle_Shapefile(modli,fname);

% This loop forms the circularly shifted and linearly increasing NxN
% channel matrix (chan). Channels are in the rows and sub-channels are
in
% the columns
chan = [];
for i = 1:N
    hshift = [0:-1:-N+1]+i-1;
    if i > 1
        hshift = [hshift(i:max(size(hshift))) hshift(1:i-1)];
    end
    chan(i,:) = hshift;
end

% This loop computes binomial coefficients to determine how many
% combinations are in each case.  The result put into a 2^N x N
% matrix of 1's and 0's called sortedBin
binChar = dec2bin([0:2^N-1]',N);
pad = repmat(' ',max(size(binChar)),1);
for i = 1:N
    binChar_pad(:,2*i) = pad;
    binChar_pad(:,2*i-1) = binChar(:,i);
end
binNum = str2num(char(binChar_pad));
binNum(:,N+1) = sum(binNum,2);
sortedBin = sortrows(binNum,[N+1 1:N]);

% This loop computes the combination number for each case
% (Case notation 2nd digit)
for i = 0:N
    [I,J,V] = find(sortedBin(:,N+1)==i);
    V = cumsum(V);
    sortedBinCol(I) = V;
end
sortedBin(:,N+2) = sortedBinCol';

% This loop uses the CRT to find all Case N1X COR, which are the
% fundamental COR shifts for all other cases and sub-cases
% The vector holding the COR is called COR, and the vector redundancyID
% contains the case label of the corresponding COR
COR = [];
redundancyID = [];
for subchan = 0:N-1
    ch = chan(:,subchan+1)';
    m = max(ch);
    a = (m-ch)/N;
    center_of_redundancy = crt(a,modli)*N-m;
    if center_of_redundancy < 0
        center_of_redundancy = center_of_redundancy + N*prod(modli);
    end
    COR = [COR center_of_redundancy];
    redundancyID = [redundancyID ...
            str2num([num2str(N),'1',num2str(subchan)])];
```

```
end

COR
redundancyID

% This loop computes the Case N1X redundancies and adds them to the
vector
% called dyn_ranges, which will be used to compute the zones of maximum
% possible dynamic ranges in which to search for the dynamic range
[CORtemp,index] = sort(COR);
CORtemp = [CORtemp Pf/2];
redundancyID = [redundancyID(index) str2num([num2str(N),'10'])];
for i = 1:length(CORtemp)-1
    dyn_ranges = [dyn_ranges; [CORtemp(i)-N+1  redundancyID(i) ...
                CORtemp(i+1)+N-1 redundancyID(i+1) ...
                CORtemp(i+1)+2*N-CORtemp(i)-1]];
end

dyn_ranges_loop = flipud(sortrows(dyn_ranges,5));

% The following statements set the maximum and minimum dynamic range
based
% on the current moduli set and the fact that the minimum dyanamic
range
% for an n-channel RSNS is always larger than the minimum dynamic range
for
% an (n-1)-channel RSNS.

% new algorithm steps
m_i = sortedBin(:,1:N).*fliplr(repmat(modli,max(size(binNum)),1));
%m_i = sortedBin(:,1:N).*(repmat(modli,max(size(binNum)),1)); %Wrong
j=find(m_i==0);
m_i(j) = 1;
prod_m_i = prod(m_i,2);
prod_m_j = prod(modli)./prod_m_i;

redundancy_len = 2*N*prod_m_j;
case_max_dynrange = N*prod_m_i+redundancy_len-1;

case_max_dynrange(2^N) = dyn_ranges_loop(1,5);

[C,I] = min(case_max_dynrange);

MAX_DYNRANGE = C(1,1)
max_dynrange_case =
str2num([num2str(sortedBin(I(1,1),N+1)),num2str(sortedBin(I(1,1),N+2)),
'0']);

[I,J] = find(redundancy_len<MAX_DYNRANGE);

sortedBin(:,N+3) = prod_m_i;
sortedBin(:,N+4) = prod_m_j;
sortedBin(:,N+5) = redundancy_len;
```

```
sortedBin(:,N+6) = case_max_dynrange;
sortedBin(I,N+7) = 1;

%MIN_DYNRANGE_limits = [1 1 42 116 419 1615]; % values are from
previous runs

%if N > length(MIN_DYNRANGE_limits)
%   MIN_DYNRANGE = MIN_DYNRANGE_limits(length(MIN_DYNRANGE_limits));
%else
%    MIN_DYNRANGE = MIN_DYNRANGE_limits(N);
%end

% This loop calculates all redundancies for all cases by calling the
% function CalculateRedundancies and stores them in a matrix
redundancy_vector = [];
for i = 1:length(I)
    redundancy_vector = [redundancy_vector; ...
            CalculateRedundancies(COR,sortedBin(I(i),:),modli)];
end

% These statements sort the redundancies and removes redundant COR.
% The first sorts the vectors by increasing COR then sorts
% by increasing Start. The next finds the unique COR with the largest
Start,
% which chooses the "smallest" redundancy for each COR (the others do
not
% affect the Dynamic Range.  The next keeps only those vectors found in
the
% previous step.  The last statement adds a counter column to the
matrix.
temp_vector = sortrows(redundancy_vector,[2 1]);

[B,I,J] = unique(temp_vector(:,2));

redundancy_vector = temp_vector(I,:)

%redundancy_vector = [redundancy_vector
[1:length(redundancy_vector)]'];

% This next section searches through the matrix of redundancies to find
% consecutive redundancies.  Once consecutive redundancies are found,
the
% number of vectors between the endpoints of the redundancies is a
% potential dynamic range.  The largest string of such vectors is the
% dynamic range.  The search routine finds all intervals and then
chooses
% the largest one(s) as the dynamic range.

% find the first redundancy
%interval_start = redundancy_vector(1,:);
%start_pointer = 2;
```

```
% loop through all of the redundancies and compute the distance bewteen
the
% start and end points of consecutive redundancies (intervals)
% which are all potential dynamic ranges

len = size(redundancy_vector,1);
redundancy_vector = sortrows(redundancy_vector,3);
diff_vector = redundancy_vector(1:len-1,1)-redundancy_vector(2:len,1);
f = find(diff_vector>=0);

while isempty(f)~=1

    redundancy_vector(f+1,:)=[];
    len = length(redundancy_vector);

    diff_vector = redundancy_vector(1:len-1,1)-
redundancy_vector(2:len,1);
    f = find(diff_vector>=0);
end

%intervals = redundancy_vector;

% This section computes the sizes of the intervals and keeps only the
% largest intervals which are the dynamic ranges
len = length(redundancy_vector);
interval_sizes = [redundancy_vector(1:len-1,1)+1
redundancy_vector(1:len-1,4:end) ...
        redundancy_vector(2:len,3)-1 redundancy_vector(2:len,4:end)];
interval_sizes(:,end+1) = (redundancy_vector(2:len,3)-1)-
(redundancy_vector(1:len-1,1)+1)+1;

dynamic_range1 = interval_sizes(find(interval_sizes(:,end)== ...
    max(interval_sizes(:,end))),:)

[rows,cols] = size(dynamic_range1);
dynamic_range = [dynamic_range1 repmat([MAX_DYNRANGE
max_dynrange_case],rows,1)];

% generating a shapefile for plotting all dynamic ranges
dynamic_range_vector = [dynamic_range1(:,1) dynamic_range1(:,3)
repmat(000,size(dynamic_range1,1),1)]

%Generate_CircleArc_Shapefile(dynamic_range_vector,modli,'_DynamicRange
');
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. MATLAB CODE TO ENCODE RSNS SIGNALS USING THE COMPUTED THRESHOLD VALUES

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%
% This program calculates comparator threshold values for a 3-Channel RSNS
% ADC system with moduli 7, 8 & 9.
%
% It then generates an analog waveform for each channel (with appropriate
% signal shift) to align them according to the starting index of the RSNS
% dynamic range.
%
% The analog waveforms are then compared with the threshold values &
% quantized into thermometer codes.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
% Authors:        Kee Leong TONG                Date: 29 Jul 2010
%                 Han Wei LIM
% Electrical & Computer Engineering Department
% Naval Postgraduate School
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
close all;

% Variable Declaration
V=1.25;           % Full-Scale Voltage
N=3;              % Number of Channels
range=128;        % Dynamic Range for Moduli 7, 8, 9
t= 0:pi/range:pi;  % Time Index (V_pi defined as a half-swing from 0 to pi)

% Moduli Values
m0=7;
m1=8;
m2=9;

% Period Calculation for each channel
P0=2*N*m0;
P1=2*N*m1;
P2=2*N*m2;

% Comparator Threshold Calculations
% Channel 1 (Mod 7)
k0=0:1:m0-1;
T0=V.*(cos(pi/2+pi.*(k0.*N+1)/P0+pi/2/P0)).^2;

% Channel 2 (Mod 8)
k1=0:1:m1-1;
T1=V.*(cos(pi/2+pi.*(k1.*N+1)/P1+pi/2/P1)).^2;

% Channel 3 (Mod 9)
```

```
k2=0:1:m2-1;
T2=V.*(cos(pi/2+pi.*(k2.*N+1)/P2+pi/2/P2)).^2;

% Generate analog waveform for each channel (with signal shift)
%mod7 = V*cos(range/P0*t).^2;
%mod8 = V*cos(range/P1*t).^2;
%mod9 = V*cos(range/P2*t).^2;
mod7 = V*cos(range/P0*t - 19*pi/2/range).^2;
mod8 = V*cos(range/P1*t - 60*pi/2/range).^2;
mod9 = V*cos(range/P2*t + 22*pi/2/range).^2;

% Generate arbituary thermometer code for each channel
mod7code = ones(1,range);
mod8code = ones(1,range);
mod9code = ones(1,range);

% Quantize Analog Waveform to Thermometer Code (Mod 7)
for i = 1:range
   if mod7(i) > T0(7)
      mod7code(i) = 7;
   else if mod7(i) > T0(6)
         mod7code(i) = 6;
      else if mod7(i) > T0(5)
            mod7code(i) = 5;
         else if mod7(i) > T0(4)
               mod7code(i) = 4;
            else if mod7(i) > T0(3)
                  mod7code(i) = 3;
               else if mod7(i) > T0(2)
                     mod7code(i) = 2;
                  else if mod7(i) > T0(1)
                        mod7code(i) = 1;
                     else
                        mod7code(i) = 0;
                     end
                  end
               end
            end
         end
      end
   end
end

% Quantize Analog Waveform to Thermometer Code (Mod 8)
for k = 1:range
   if mod8(k) > T1(8)
      mod8code(k) = 8;
   else if mod8(k) > T1(7)
         mod8code(k) = 7;
      else if mod8(k) > T1(6)
            mod8code(k) = 6;
         else if mod8(k) > T1(5)
               mod8code(k) = 5;
            else if mod8(k) > T1(4)
```

```
                mod8code(k) = 4;
            else if mod8(k) > T1(3)
                mod8code(k) = 3;
            else if mod8(k) > T1(2)
                mod8code(k) = 2;
            else if mod8(k) > T1(1)
                mod8code(k) = 1;
            else
                mod8code(k) = 0;
            end
        end
    end
  end
end

% Quantize Analog Waveform to Thermometer Code (Mod 9)
for l = 1:range
  if mod9(l) > T2(9)
    mod9code(l) = 9;
  else if mod9(l) > T2(8)
      mod9code(l) = 8;
    else if mod9(l) > T2(7)
        mod9code(l) = 7;
      else if mod9(l) > T2(6)
          mod9code(l) = 6;
        else if mod9(l) > T2(5)
            mod9code(l) = 5;
          else if mod9(l) > T2(4)
              mod9code(l) = 4;
            else if mod9(l) > T2(3)
                mod9code(l) = 3;
              else if mod9(l) > T2(2)
                  mod9code(l) = 2;
                else if mod9(l) > T2(1)
                    mod9code(l) = 1;
                else
                    mod9code(l) = 0;
                end
              end
            end
          end
        end
      end
    end
  end
end

% Generate Threshold Value indices for each channel
t71(1:range)=T0(1);
```

```
t72(1:range)=T0(2);
t73(1:range)=T0(3);
t74(1:range)=T0(4);
t75(1:range)=T0(5);
t76(1:range)=T0(6);
t77(1:range)=T0(7);

figure(1)
subplot(2,1,1)
hold on
plot(mod7,'b-')
plot(t71,'k--')
plot(t72,'k--')
plot(t73,'k--')
plot(t74,'k--')
plot(t75,'k--')
plot(t76,'k--')
plot(t77,'k--')
title('Channel 1 (Modulus 7)')
xlabel('Index')
ylabel('Analog Voltage Levels')

subplot(2,1,2)
plot(mod7code)
title('Thermometer Code')
xlabel('Index')
ylabel('Number of Comparators On')

t81(1:range)=T1(1);
t82(1:range)=T1(2);
t83(1:range)=T1(3);
t84(1:range)=T1(4);
t85(1:range)=T1(5);
t86(1:range)=T1(6);
t87(1:range)=T1(7);
t88(1:range)=T1(8);

figure(2)
subplot(2,1,1)
hold on;
plot(mod8)
plot(t81,'k--')
plot(t82,'k--')
plot(t83,'k--')
plot(t84,'k--')
plot(t85,'k--')
plot(t86,'k--')
plot(t87,'k--')
plot(t88,'k--')
title('Channel 2 (Modulus 8)')
xlabel('Index')
ylabel('Analog Voltage Level')

subplot(2,1,2)
```

```
plot(mod8code)
title('Thermometer Code')
xlabel('Index')
ylabel('Number of Comparators On')

t91(1:range)=T2(1);
t92(1:range)=T2(2);
t93(1:range)=T2(3);
t94(1:range)=T2(4);
t95(1:range)=T2(5);
t96(1:range)=T2(6);
t97(1:range)=T2(7);
t98(1:range)=T2(8);
t99(1:range)=T2(9);

figure(3)
subplot(2,1,1)
hold on
plot(mod9)
plot(t91,'k--')
plot(t92,'k--')
plot(t93,'k--')
plot(t94,'k--')
plot(t95,'k--')
plot(t96,'k--')
plot(t97,'k--')
plot(t98,'k--')
plot(t99,'k--')
title('Channel 3 (Modulus 9)')
xlabel('Index')
ylabel('Analog Voltage Level')

subplot(2,1,2)
plot(mod9code)
title('Thermometer Code')
xlabel('Index')
ylabel('Number of Comparators On')
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. MATLAB CODE TO PLOT THE QUANTIZED VALUES AND COMPUTE THE STEP SIZE, DNL AND INL

```
% PROCESS TRANSFER FUNCTION DATA - RSNS
% M^ = 43 - 3 = 40
clear all;
close all;

load onekhztriangle_2.txt
trans=onekhztriangle_2([74:124],1)-min(onekhztriangle_2([74:124],1));  %
Store one cycle of dynamic range & normalise to start from zero.
[rr,cc]=size(trans);
rrs=1:1:rr;
transb=dec2bin(trans);
transbc=str2num(transb);
LSB=0.102;              % Effective V_pi/(N*m_i)

% Convert the signal to analog
Quantized_Data = (trans).*(3.22/39);

% Step Size
step_size = hist(Quantized_Data(1:51),40);
step_size= step_size/1.1;      % Convert to LSBs

% DNL
actual_dnl = step_size - 1;

% Process INL
inl(1)=actual_dnl(1);
for ij=2:40
    inl(ij)=inl(ij-1)+actual_dnl(ij);
end

QD=Quantized_Data(1:51);
sample_index=0:50;

ideal_Quantized_Data = 0:3.22/40:3.22;
ideal_sample_index = 0:length(ideal_Quantized_Data)-1;

% Quantization error
idea=[LSB/2:LSB:4.5]';
v=sample_index*(3.22/40);
nums=hist(QD,40);
trans=[v' QD];
ct=1;
for ii=1:40
     for jj=1:nums(1,ii)
         qerror(ct,1)=idea(ii,1)-trans(ct,1);
         ct=ct+1;
     end
end
```

```
% Plot Transfer Function
figure(1)
stairs(sample_index*(3.22/40), QD');
title('ADC Transfer Function');
xlabel('Input Voltage (V)');
ylabel('Decimal Output');
grid on

figure(2)
plot(sample_index*(3.22/40), qerror');
xlabel('Input Voltage (V)');
ylabel( 'Quantization Error');
axis([0 4.5 -0.15 0.2]);
grid on;

 % Plot Ideal Transfer Function
figure(3)
hold on
stairs(ideal_sample_index*(3.22/40), ideal_Quantized_Data,'r');
title('ADC Transfer Function');
%xlabel('Input Voltage (V)');
ylabel( 'Binary Code');
grid on

ac=0:1:39;
acc=ac*LSB;

% Plot Step Size as a function of input voltage
figure(3)
subplot(3,1,1)
hold on
stem(acc, step_size);
ylabel('LSB');
title('Step Size');
grid on

%Plot DNL
hold on
subplot(3,1,2)
stem(acc,actual_dnl);
axis([0 4 -1 2]);
title('DNL');
ylabel('LSB');
grid on

%Plot INL
hold on
subplot(3,1,3)
stem(acc,inl);
axis([0 4 -1 8])
title('INL');
xlabel('Input Voltage(V)');
ylabel('LSB');
```

APPENDIX E. MATLAB CODE TO COMPUTE THE ADC SPECTRAL AVERAGE AND DYNAMIC PARAMETERS

```
% 1-kHz sine wave input signal
% Read data from file
clear all; close all;
nbis = 6;
m = 2;          % specify number of columns
num_samples = 4096;     % number of samples
fsample = 100e3;
T = 1/fsample;
fo = 1000;
time = 0:T:(num_samples-1)*T;
freq = [0:num_samples/2 - 1].*fsample/num_samples;

% Create cells for data
Binary_Data = zeros(num_samples,20);
Quantized_Data = zeros(num_samples,20);
sig_fft = zeros(num_samples);
sig_blackmanharris = zeros(num_samples,20);
sig_fft_blackmanharris = zeros(num_samples,20);
Data_Array = zeros(num_samples,20);

% read and process 20 sets of sampled data
for i = 1:20
    a = 'Run';
    b = int2str(i);
    c = '.lvm';
    filename = strcat(a,b,c);
    fid = fopen(filename);
    Data = textscan(fid, '%14f%14f', num_samples);
    Data(:,[1])=[]; % Remove column 1
    Data_cell_array{i,:} = Data{1,:};
    Data_Sc = Data{1,:};
    Data_Array = horzcat(Data_Sc, Data_Array);

% Convert the signal to analog
Quantized_Data(1:num_samples,i) = (Data_Array(:,i)-1).*(3.22/41);

% Use BlackmanHarris/Hamming/Hanning Window
sig_blackmanharris(1:num_samples,i) =
Quantized_Data(1:num_samples,i).*blackmanharris(num_samples);
%sig_hanning = Quantized_Data.*hanning(num_samples);
%sig_hamming = Quantized_Data.*hamming(num_samples);

% Perform FFT on the signals
sig_fft = abs(fft(Quantized_Data(:,1), num_samples)/num_samples);
sig_fft_blackmanharris(1:num_samples,i) =
abs(fft(sig_blackmanharris(:,i), num_samples)/num_samples);

end
```

```
%calculate mean fft value
ave_fft_blackmanharris = mean(sig_fft_blackmanharris');

%calculate magnitude squared spectrum
spectP_sig = (sig_fft.*sig_fft);
spectP_sig_blackmanharris =
ave_fft_blackmanharris.*ave_fft_blackmanharris;

% find maximum other than DC
max_sig = max(sig_fft(10:num_samples/2));
max_blackmanharris = max(ave_fft_blackmanharris(10:num_samples/2));

% normalize mean data
%ave_norm_fft_sig = ave_fft_sig./max_sig;
ave_norm_fft_blackmanharris =
ave_fft_blackmanharris./max_blackmanharris;

% Plot Signals from normalized blackmanharris window
figure(1)
plot(freq/500, db(ave_norm_fft_blackmanharris(1:num_samples/2),
'voltage'));
title('Normalized Magnitude Squared Spectrum Using Blackman-Harris
Window');
xlabel('Frequency(kHz)');
ylabel('Magnitude(dB)');
grid on
hold on

% Plot Signals magnitude spectrum with window
figure(2)
plot(freq/500, db(2*ave_fft_blackmanharris(1:num_samples/2),
'voltage'));
title('Frequency Spectrum Plot Using Blackman-Harris Window');
xlabel('Frequency(kHz)');
ylabel('Amplitude(V)');
grid on
hold on

% magnitude spectrum without window
figure(3)
plot(freq/500, db(2*sig_fft(1:num_samples/2),'voltage'), 'r');
title('Frequency Spectrum Plot Without Window');
xlabel('Frequency(kHz)');
ylabel('Amplitude(dB)');
grid on
hold on

% power spectrum with and without spectral averaging
figure(4)
plot(freq/500, db(spectP_sig(1:num_samples/2), 'power'),'r', freq/500,
db(spectP_sig_blackmanharris(1:num_samples/2), 'power'));
title('Power Spectrum');
xlabel('Frequency(kHz)');
ylabel('Power(dB)');
```

```
legend('Signal', 'Spectral Average Using Blackman-Harris Window');
grid on
hold on

% power spectrum with window
figure(5)
plot(freq/500, db(spectP_sig_blackmanharris(1:num_samples/2),
'power'));
title('Power Spectrum Plot Using Blackman-Harris Window');
xlabel('Frequency(kHz)');
ylabel('Power(dB)');
grid on
hold on

ADC=3.5/2+1.65*sin(2*pi*fo.*time-3);

figure(6)
plot(time(1:120),ADC(1:120),'b-',time(1:120),Quantized_Data(1:120),'r-
');
title('ADC 1kHz Input Signal vs FPGA Output Signal');
xlabel('Time(sec)');
ylabel('Amplitude(V)');
legend('ADC Input Signal','FPGA Output Signal');
grid on

%calculate noise floor
noise_floor_M2 = 10*log10(3*num_samples/8) + 6.02*log2(41)
noise_floor_MC = 10*log10(num_samples/2) + 1.76 + 6.02*log2(41)
noise_floor_MJ = 20*log10(1/(2*pi*std(ave_fft_blackmanharris))) +
20*log10(fsample/fo) + 10*log10(num_samples/8)

% Calculate SNR, SINAD, ENOB, THD and SFDR
fin_sig = find(sig_fft(1:num_samples/2) == max_sig);
fin_blackmanharris = find(ave_fft_blackmanharris(1:num_samples/2) ==
max_blackmanharris);

% Span of the input frequency on each side
span=max(floor(num_samples/250),5);

% Approximate search span for harmonics on each side
spanh=40;

% Determine power spectrum
%spectP=(sig_fft).*(sig_fft);
%figure(6)
%plot(freq/500, db(spectP(1:num_samples/2)));
%title('Power Spectrum Plot');
%xlabel('Frequency(kHz)');
%ylabel('Power(dB)');
%grid on

% Find DC offset power
Pdc=(spectP_sig_blackmanharris(1));
% Extract overall signal power
```

```
%Ps=sum(spectP(fin));

% Vector/matrix to store both frequency and power of signal and
harmonics
Fh=[];

% The 1st element in the vector/matrix represents the signal, the next
element represents % the 2nd
% harmonic, etc.
Ph = [];
Ph_sig = [];
sort_harmonics = sort(spectP_sig_blackmanharris, 'descend');

for num_harmonics = 0:4
    % Extract harmonics
    har_bin = fin_blackmanharris +
(num_harmonics*2*(fin_blackmanharris-1));
    Ph=[Ph spectP_sig_blackmanharris(har_bin)];
    har_bin_sig = fin_sig + (num_harmonics*2*(fin_sig-1));
    Ph_sig=[Ph_sig spectP_sig(har_bin)];
end

% Determine Total Distortion Power and Noise Power
P_thd = (Ph(2:5));
P_thd_sig = (Ph_sig(2:5));

%Pn = sum(spectP(1:num_samples/2)) - Pdc - Ps - Pd;

%Calculate dynamic parameters
SNR_db = 6.02*log2(43) + 1.76
THD_db = 20*log10(sqrt(sum(P_thd)))
THD_sig_db = 20*log10(sqrt(sum(P_thd_sig)))
SINAD_db = -20*log10(sqrt((10^-(SNR_db/10) + 10^(THD_db/10))))
SINAD_sig_db = -20*log10(sqrt((10^-(SNR_db/10) + 10^(THD_sig_db/10))))
ENOB = (SINAD_db - 1.76)/6.0206
ENOB_sig = (SINAD_sig_db - 1.76)/6.0206
SFDR_dB = 10*log10(Ph(1)/max(Ph(2:5)))
%SNR_dB = 10*log10(Ps/Pn)
%SINAD_dB = 10*log10(Ps/(Pn+Pd))
%THD_dB = 10*log10(Pd/Ph(1))

%HD = 10*log10(Ph(1:11)/Ph(1));

%plot(Data_Array);
fid = fclose('all');
```

```
% 2- kHz sine wave input signal.
% Read data from file
clear all;
close all;
nbits = 6;
m = 2;          % specify number of columns
num_samples = 4096;     % number of samples
fsample = 100e3;
T = 1/fsample;
fo = 2000;
time = 0:T:(num_samples-1)*T;
freq = [0:num_samples/2 - 1].*fsample/num_samples;

% Create cells for data
Binary_Data = zeros(num_samples, 20);
Quantized_Data = zeros(num_samples, 20);
sig_fft = zeros(num_samples);
sig_blackmanharris = zeros(num_samples,20);
sig_fft_blackmanharris = zeros(num_samples,20);
Data_Array = zeros(num_samples, 1);

% read and process 20 sets of sampled data
for i = 1:20
    a = 'Run';
    b = int2str(i);
    c = '.lvm';
    filename = strcat(a,b,c);
    fid = fopen(filename);
    Data = textscan(fid, '%14f%14f', num_samples);
    % Data(:,[1])=[]; % Remove column 1
    Data_cell_array{i,:} = Data{1,:};
    Data_Sc = Data{1,:};
    Data_Array = horzcat(Data_Sc, Data_Array);

% Convert the signal to analog
Quantized_Data(1:num_samples,i) = (Data_Array(:,i)-1).*(3.22/41);

% Use BlackmanHarris/Hamming/Hanning Window
sig_blackmanharris(1:num_samples,i) =
Quantized_Data(1:num_samples,i).*blackmanharris(num_samples);
%sig_hanning = Quantized_Data.*hanning(num_samples);
%sig_hamming = Quantized_Data.*hamming(num_samples);

% Perform FFT on the signals
sig_fft = abs(fft(Quantized_Data(:,1), num_samples)/num_samples);
sig_fft_blackmanharris(1:num_samples,i) =
abs(fft(sig_blackmanharris(:,i), num_samples)/num_samples);

end

%calculate mean fft value
ave_fft_blackmanharris = mean(sig_fft_blackmanharris');

%calculate magnitude squared spectrum
```

```
spectP_sig = (sig_fft.*sig_fft);
spectP_sig_blackmanharris =
ave_fft_blackmanharris.*ave_fft_blackmanharris;

% find maximum other than DC
max_sig = max(sig_fft(10:num_samples/2));
max_blackmanharris = max(ave_fft_blackmanharris(10:num_samples/2));

% normalize mean data
%ave_norm_fft_sig = ave_fft_sig./max_sig;
ave_norm_fft_blackmanharris =
ave_fft_blackmanharris./max_blackmanharris;

% Plot Signals from normalized blackmanharris window
figure(1)
plot(freq/500, db(ave_norm_fft_blackmanharris(1:num_samples/2),
'voltage'));
title('Normalized Magnitude Squared Spectrum Using Blackman-Harris
Window');
xlabel('Frequency(kHz)');
ylabel('Magnitude(dB)');
grid on
hold on

% Plot Signals magnitude spectrum with window
figure(2)
plot(freq/500, db(2*ave_fft_blackmanharris(1:num_samples/2),
'voltage'));
title('Frequency Spectrum Plot Using Blackman-Harris Window');
xlabel('Frequency(kHz)');
ylabel('Amplitude(V)');
grid on
hold on

% magnitude spectrum without window
figure(3)
plot(freq/500, db(2*sig_fft(1:num_samples/2),'voltage'), 'r');
title('Frequency Spectrum Plot Without Window');
xlabel('Frequency(kHz)');
ylabel('Amplitude(dB)');
grid on
hold on

% power spectrum with and without spectral averaging
figure(4)
plot(freq/500, db(spectP_sig(1:num_samples/2), 'power'),'r', freq/500,
db(spectP_sig_blackmanharris(1:num_samples/2), 'power'));
title('Power Spectrum');
xlabel('Frequency(kHz)');
ylabel('Power(dB)');
legend('Signal', 'Spectral Average Using Blackman-Harris Window');
grid on
hold on
```

```
% power spectrum with window
figure(5)
plot(freq/500, db(spectP_sig_blackmanharris(1:num_samples/2),
'power'));
title('Power Spectrum Plot Using Blackman-Harris Window');
xlabel('Frequency(kHz)');
ylabel('Power(dB)');
grid on
hold on

figure(6)
plot((0:100)*T, Quantized_Data(1:101));
title('Ramp Function with 2kHz Frequency');
xlabel('Time(sec)');
ylabel('Amplitude(V)');
grid on

%calculate noise floor
noise_floor_M2 = 10*log10(3*num_samples/8) + 6.02*log2(41)
noise_floor_MC = 10*log10(num_samples/2) + 1.76 + 6.02*log2(41)
noise_floor_MJ = 20*log10(1/(2*pi*std(ave_fft_blackmanharris))) +
20*log10(fsample/fo) + 10*log10(num_samples/8)

% Calculate SNR, SINAD, ENOB, THD and SFDR
fin_sig = find(sig_fft(1:num_samples/2) == max_sig);
fin_blackmanharris = find(ave_fft_blackmanharris(1:num_samples/2) ==
max_blackmanharris);

% Span of the input frequency on each side
span=max(floor(num_samples/250),5);

% Approximate search span for harmonics on each side
spanh=40;

% Determine power spectrum
%spectP=(sig_fft).*(sig_fft);
%figure(6)
%plot(freq/500, db(spectP(1:num_samples/2)));
%title('Power Spectrum Plot');
%xlabel('Frequency(kHz)');
%ylabel('Power(dB)');
%grid on

% Find DC offset power
Pdc=(spectP_sig_blackmanharris(1));

% Extract overall signal power
%Ps=sum(spectP(fin));

% Vector/matrix to store both frequency and power of signal and
harmonics
Fh=[];
```

```
% The 1st element in the vector/matrix represents the signal, the next
element represents % the 2nd
% harmonic, etc.
Ph = [];
Ph_sig = [];
sort_harmonics = sort(spectP_sig_blackmanharris, 'descend');

for num_harmonics = 0:4
    % Extract harmonics
    har_bin = fin_blackmanharris +
(num_harmonics*2*(fin_blackmanharris-1));
    Ph=[Ph spectP_sig_blackmanharris(har_bin)];
    har_bin_sig = fin_sig + (num_harmonics*2*(fin_sig-1));
    Ph_sig=[Ph_sig spectP_sig(har_bin)];
end

% Determine Total Distortion Power and Noise Power
P_thd = (Ph(2:5));
P_thd_sig = (Ph_sig(2:5));

%Pn = sum(spectP(1:num_samples/2)) - Pdc - Ps - Pd;

%Calculate dynamic parameters
SNR_db = 6.02*log2(41) + 1.76
THD_db = 20*log10(sqrt(sum(P_thd)))
THD_sig_db = 20*log10(sqrt(sum(P_thd_sig)))
SINAD_db = -20*log10(sqrt((10^-(SNR_db/10) + 10^(THD_db/10))))
SINAD_sig_db = -20*log10(sqrt((10^-(SNR_db/10) + 10^(THD_sig_db/10))))
ENOB = (SINAD_db - 1.76)/6.0206
ENOB_sig = (SINAD_sig_db - 1.76)/6.0206
SFDR_dB = 10*log10(Ph(1)/max(Ph(2:5)))
%SNR_dB = 10*log10(Ps/Pn)
%SINAD_dB = 10*log10(Ps/(Pn+Pd))
%THD_dB = 10*log10(Pd/Ph(1))

%HD = 10*log10(Ph(1:11)/Ph(1));

%plot(Data_Array);
%fid = fclose('all');
```

LIST OF REFERENCES

[1] H. F. Taylor, "An optical analog-to-digital converter – design and analysis," *IEEE J. Quantum Electron*, Vol. 15, pp. 210–216, Apr 1979.

[2] R. Becker and F. Leonberger, "2-bit 1 Gsample/s electrooptic guided-wave analog-to-digital converter," *IEEE Journal of Quantum Electronics*, Vol. 18, No. 10, pp. 1411–1413, Oct 1982.

[3] A. S. Bhushan, P. Kelkar and B. Jalali, "30 Gsample/s time-stretch analogue-to-digital converter," *Electronics Letters*, Vol. 36, No. 18, pp. 1526–1527, 31 Aug 2000.

[4] H. Yan and B. Jalali, "Photonic time-stretch analog-to-digital converter: fundamental concepts and practical considerations," *Journal of Lightwave Technology*, Vol. 21, No.12, pp. 3085–3103, Dec 2003.

[5] G. C. Valley, G. A. Sefler, C. Chou and B. Jalali, "Continuous time realization of time-stretch ADC," *International Topical Meeting on Microwave Photonics, 2006.* MWP 2006*,* pp. 1–3, Oct 2006.

[6] R. Urata, L. Y. Nathawad, R. Takahashi, Kai Ma, D. A. B. Miller, B. A. Wooley and J. S. Harris Jr., "Photonic A/D conversion using low-temperature-grown GaAs MSM switches integrated with Si-CMOS," *Journal of Lightwave Technology*, Vol. 21, No. 12, pp. 3104–3115, Dec 2003.

[7] R. Urata, R. Takahashi, V. A. Sabnis, D. A. B. Miller, and J. S. Harris Jr., "Ultrafast differential sample and hold using low-temperature-grown GaAs MSM for photonic A/D conversion," *IEEE Photonics Technology Letters*, Vol. 13, No. 7, pp. 717–719, Jul 2001.

[8] R. Urata, L. Y. Nathawad, Kai Ma, R. Takahashi, D. A. B. Miller, B. A. Wooley and J. S. Harris Jr., "Ultrafast sampling using low temperature grown GaAs MSM switches integrated with CMOS amplifier for photonic A/D conversion," *The 15th Annual Meeting of the IEEE Lasers and Electro-Optics Society, 2002*. LEOS 2002. Vol. 2, pp. 809-810, 10–14 Nov 2002.

[9] R. Urata, R. Takahashi, V. A. Sabnis, D. A. B. Miller, and J. S. Harris, "High-speed sample and hold using low temperature grown GaAs MSM switches for photonic A/D conversion," *Summaries of papers presented at the Conference on Lasers and Electro-Optics, 2001*. CLEO '01. Technical Digest, pp. 66–67, 2001.

[10] I. H. Wang and S. I. Liu, "A CMOS 5-bit 5 Gsample/sec analog-to-digital converter in 0.13 um CMOS," *Journal of Semiconductor Technology and Science,* Vol. 7, No. 1, pp. 28–35, Mar 2007.

[11] G. C. Valley, "Photonic analog-to-digital converters," *Optics Express,* Vol. 15, No. 5, pp. 1955–1982, 5 Mar 2007.

[12] T. R. Clark, J. U. Kang and R. D. Esman, "Performance for a time and wavelength-interleaved photonic sampler for analog-to-digital conversion," *Photonics Technology Letters, IEEE*, Vol. 11, No. 9, pp. 1168–1170, Sep 1999.

[13] J. C. Twitchell, J. L. Wasserman, P. W. Juodawlkis, G. E. Betts and R. C. Williamson, "High-linearity 208 MS/s photonic analog-to-digital converter using 1-to-4 optical time-division multiplexing," *IEEE Photonics Technology Letters,* Vol. 13, No. 7, pp.714–716, Jul 2001.

[14] H. Zmuda, E. N. Toughlian, G. Li and P. LiKamWa, "A photonic wideband analog-to-digital converter," *Aerospace Conference, 2001*, IEEE Proceedings. Vol. 3, pp.3/1461-3/1472, 2001.

[15] E. N. Toughlian and H. Zmuda, "A photonic wide-band analog to digital converter," *International Topical Meeting on Microwave Photonics 2000*, MWP 2000, pp. 248–250, 2000.

[16] S. Oda and A. Maruta, "A novel quantization scheme by slicing supercontinuum spectrum for all-optical analog-to-digital conversion," *IEEE Photonics Technology Letters,* Vol. 17, No. 2, pp. 465–467, Feb 2005.

[17] C. W. Holzwarth, R. Amatya, M. Araghchini, J. Birge, H. Byun, J. Chen, M. Dahlem, N. A. DiLello, F. Gan, J. L.Hoyt, E. P. Ippen, F. X. Kartner, A. Khilo, J. Kim, M. Kim, A. Motamedi, J. S. Orcutt, M. Park, M. Perrott, M. A. Popovic, R. J. Ram, H. I. Smith, G. R. Zhou, S. J.Spector, T. M. Lyszczarz, M. W. Geis, D. M. Lennon, J. U. Yoon, M. E. Grein, R. T. Schulein, S. Frolov, A. Hanjani and J. Shmulovich, "High speed analog-to-digital conversion with silicon photonics," *SPIE Proceedings 7220*, Vol. 72200B, pp. 1–15, 2009.

[18] P. E. Pace, D. Styer and I. A. Akin, "A folding ADC preprocessing architecture employing a robust symmetrical number system with gray-code properties," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 47, No. 5, pp. 462–467, May 2000.

[19] P. E. Pace, R. E. Walley, R. J. Pieper and J. P. Powers, "5bit guided-wave SNS transfer characteristics," *Electronics Letters*, Vol 31, pp. 1799–1800, Oct 1995.

[20] P. E. Pace, J. P. Powers, R. J. Pieper, R. Walley, H. Yamakoshi, C. Crowe and B. Nimri, "8-bit integrated optical SNS ADC," *Proceedings of the Twenty-Seventh Southeastern Symposium on System Theory*, pp. 144–148, Mar 1995.

[21] P. E. Pace, D. C. Jenn and J. P. Powers, "Symmetrical number systems: Theory and applications," Transworld Research Network.

[22] P. E. Pace, J. Schafer and D. Styer, "Optimum analog preprocessing for folding ADCs," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 42, pp. 825–829, 1995.

[23] P. E. Pace, P. Ramamoorthy and D. Styer, "A preprocessing architecture for resolution enhancement in high-speed analog-to-digital converters," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 41, pp. 373–379, 1994.

[24] P. E. Pace, W. Ringer, K. Foster and J. Powers, "Optical signal integrity and interpolation signal processing in wideband SNS digital antennas," *Proc. 1997 DARPA Photonic Systems for Antenna Applications,* 13 Jan 1997.

[25] P. E. Pace, R. Leino and D. Styer, "Use of the symmetrical number system in resolving single-frequency undersampling aliases," *IEEE Transactions on Signal Processing*, Vol. SP-45, pp. 1153–1160, 1997.

[26] P. E. Pace and D. Styer, "High-resolution encoding process for an integrated optical analog-to-digital converter," *Optical Engineering*, Vol. 33, pp. 2638–2645, Aug 1994.

[27] P. E. Pace, D. Wickersham, D. C. Jenn, N. S. York, "High-resolution phase sampled interferometery using symmetrical number systems," *IEEE Transactions on Antennas and Propagation*, Vol. 49, pp. 1411–1423, Oct 2001.

[28] D. C. Jenn, P. E. Pace, T. Hatziathanasiou and R. Vitale, "High resolution wideband direction finding arrays based on optimum symmetrical number system encoding," *Electronics Letters*, Vol 34, pp. 1062–1063, May 1998.

[29] D. Styer and P. E. Pace, "Two-Channel RSNS Dynamic Range," *IEEE Transactions on Signal Processing*, Vol. 49, pp. 395–397, Mar 2002.

[30] M. Arvizo, "Electro-Optic analog-to-digital converter (ADC) architecture based on the Robust Symmetrical Number System," Master's thesis, Naval Postgraduate School, Monterey, California, Sep 2009.

[31] H. W. Lim, "FPGA implementation of robust symmetrical number system in high-speed folding analog-to-digital converters," Master's thesis, Naval Postgraduate School, Monterey, California, Dec 2010.

[32] B. L. Luke, P.E. Pace, "Computation of the Robust Symmetrical Number System Dynamic Range," *IEEE Information Theory Workshop*, Aug 30–Sep 3, 2010.

[33] Sumitomo Osaka Ltd., "Application note for LN Modulators," http://www.lambdaphoto.co.uk/pdfs/SumitomoModulatorApplicationNote.pdf, accessed on 05 Mar 2010.

[34] Sumitomo Osaka Ltd., "Application note for Modulators," http://www.lambdaphoto.co.uk/pdfs/SumitomoModulatorApplicationNote.pdf, accessed on 06 Mar 2010.

[35] P. E. Pace, *Advanced Techniques for Digital Receivers*, Artech, Massachusetts, 2000.

[36] Sumitomo Osaka Ltd., "Application note for Optics Modulators," http://www.lambdaphoto.co.uk/pdfs/SumitomoModulatorApplicationNote.pdf, accessed on 08 Mar 2010.

[37] Newport Corporation, "Responsivity plot of InGaAs photodetectors," http://www.newport.com, accessed on 10 Jul 2010.

INITIAL DISTRIBUTION LIST

1. Defence Technical Information Center
   Fort Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Chairman, Code ECE
   Naval Postgraduate School
   Monterey, CA

4. Professor Phillip Pace
   Naval Postgraduate School
   Monterey, CA

5. Professor David Jenn
   Naval Postgraduate School
   Monterey, CA

6. Mr. James Calusdian
   Naval Postgraduate School
   Monterey, CA

7. Professor Yeoh Tat Soon
   National University of Singapore
   Singapore

8. Tan Lai Poh
   National University of Singapore
   Singapore